НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

«На правах рукопису»
УДК 004.056

«До захисту допущено»
В.о. завідувача кафедри
_____ М.В.Грайворонський
“___”_____2018 р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності: 125 Кібербезпека

на тему: Захист мобільних застосунків на основі систем з нульовим знанням

Виконав (-ла): студент (-ка) 2 курсу, групи ФБ-71мп
(шифр групи)

Санак Олексій Євгенійович
(прізвище, ім’я, по батькові)

Науковий керівник к.т.н., доц. Коломицев Михайло Володимирович _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____ _____ _____
(назва розділу) (науковий ступінь, вчене звання, , прізвище, ініціали) (підпис)

Рецензент к.т.н., доцента ФІОТ Пасько В.П. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2018 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

Рівень вищої освіти – другий (магістерський) за освітньо-професійною програмою
Спеціальність (спеціалізація) – 125 Кібербезпека («Системи і технології кібербезпеки)

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ М.В.Грайворонський
(підпис)

«___»_____2018 р.

ЗАВДАННЯ
на магістерську дисертацію студенту

Санаку Олексію Євгенійовичу

1. Тема дисертації: Захист мобільних застосунків на основі систем з нульовим знанням

науковий керівник дисертації к.т.н., доц. Коломицев Михайло Володимирович,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «15» листопада 2018 р. № 4171-с

2. Термін подання студентом дисертації 12.12.2018 р.

3. Об'єкт дослідження _____
_____

4. Вихідні дані _____
_____
_____
_____

5. Перелік завдань, які потрібно розробити _____
_____
_____
_____
_____
_____

6. Орієнтовний перелік ілюстративного матеріалу _____
_____
_____

7. Орієнтовний перелік публікацій _____

_____

_____

8. Консультанти розділів дисертації*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

9. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка

Студент _____ (підпис) _____ (ініціали, прізвище)

Науковий керівник дисертації _____ (підпис) _____ (ініціали, прізвище)

* Консультантом не може бути зазначено наукового керівника магістерської дисертації.

РЕФЕРАТ

Робота обсягом 121 сторінки містить 15 ілюстрацій, 25 таблиць та 7 літературних посилань.

Метою даної кваліфікаційної роботи є аналіз середовища мобільних застосунків, їх архітектури, дослідження можливих вразливостей застосунків та методів боротьби з ними.

Об'єктом дослідження є сфера розробки мобільних застосунків.

Предметом дослідження є інструменти, правила та інструкції з забезпечення захищеності мобільних застосунків.

Результати роботи викладені у вигляді схеми архітектури системи безпеки мобільного застосунку, набору критеріїв до застосовуваних інструментів захисту, правил, що повинні бути дотримані в системі безпеці, та обов'язкових складових політики безпеки, що буде застосовуватись.

Результати роботи можуть бути використані при розробці мобільних застосунків, а також для модернізації вже існуючих застосунків впровадженням запропонованої систем захисту. Також можливе використання окремих компонентів запропонованої системи безпеки та варіантів їх впровадження.

БЕЗПЕКА МОБІЛЬНИХ ПРИСТРОЇВ, БЕЗПЕКА ПРИВАТНИХ ДАНИХ, МОБІЛЬНІ ЗАСТОСУНКИ, КІБЕРБЕЗПЕКА

ABSTRACT

The work includes 121 pages, 15 illustrations, 25 tables and 7 literary references.

The purpose of this qualification work is to analyze the environment of mobile applications, their architecture, the study of possible vulnerabilities in applications and methods to combat them.

The object of research is the field of development of mobile applications.

The subject of the study is the tools, rules and instructions for ensuring the security of mobile applications.

The results of the work are presented in the form of a scheme of the architecture of the security system of the mobile application, a set of criteria for the applicable security tools, rules that must be observed in the security system, and the mandatory components of the security policy to be applied.

The results of the work can be used in the development of mobile applications, as well as for the modernization of existing applications by implementation of the proposed protection systems. It is also possible to use the individual components of the proposed security system and their implementation options.

MOBILE DEVICES SECURITY, SECURITY OF PRIVATE DATA, MOBILE APPLICATIONS, CYBERSECURITY

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

«User-friendly» дизайн – позначення інтерфейсу, що є дружнім (зрозумілим) користувачеві в теорії використання сайту чи застосунку.

Мікротранзакція – популярна бізнес-модель розповсюдження завантажуваного контенту або доступу до надаваних послуг за невеликими цінами (зазвичай в межах 10 $).

«Freemium» (укр. фрімиум) – бізнес модель, яка полягає в пропозиції скористатись безкоштовним використанням продукту, в той час, коли додатковий розширений контент (покращена преміум-версія програми, додатковий сюжет в іграх) пропонується за додаткову плату. Приклад – 30-денний пробний період використання програми.

«Hardware» – набір технічних рішень та засобів, що собою являє електронний пристрій. Приклад – складові електронно-обчислювальної машини: зовнішні пристрої, термінали, абонентські пункти, тощо, які необхідні для функціонування тієї чи іншої системи; фізична частина ЕОМ.

PWA (Progressive Web Apps) – мобільний веб-сайт, який трансформується додатковими інструментами та функціями, що дає йому остаточне відчуття "застосунку". Він відображається за допомогою Google Chrome і використовує службові сервіси для забезпечення безперебійної роботи навіть без надійного підключення до Інтернету.

IoT (Internet of Things) – це концепція мережі приладів, що мають можливість контакту один з одним, різноманітні датчики та вбудоване ПЗ для передачі та обміну даними між фізичним світом та комп’ютерними системами.

«On-demand» застосунки – застосунки на вимогу, сервіси забезпечення індивідуальних потреб користувача по типу доставки їжі, мультимедіа контенту, різноманітних робіт.

«MHealth» застосунки – мобільні застосунки, що націлені в першу чергу на збір, аналіз та зберігання даних про фізичну активність людини, медичні та фізичні показники, записи про медичний стан, показання, на допомогу у веденні лікування тих и інших хвороб.

Бекдор (англ. backdoor) – це дефект алгоритму, що дозволяє отримати несанкціонований доступ до даних користувача чи до управління системою.

ВСТУП

В наш час ринок мобільних застосунків розвивається рекордно швидкими темпами, але за прогнозами провідних компаній це ще не межа такого темпу розвитку. На ринку мобільних розваг та застосунків з'являється все більше розробників, компаній-розробників та відповідно ще більше застосунків. Дохід від реалізації продуктів, створюваних мобільною індустрією досягає незнаних раніше обсягів.

Значні темпи розвитку даної індустрії в більшій мірі завдячує різноманітності монетизації мультимедійного контенту, що відтворюється в мобільних застосунках. На першому серед таких методів отримання грошей є звичайно надокучлива реклама, потім ідуть застосунки типу «freemium», на наступному щаблі розвитку викачки грошей знаходиться вже платний застосунок і на останок це програми, що передбачають в собі мікротранзакції.

При розробці та впровадженні готових рішень розробники прибігають до самих різноманітних способів монетизації, оптимізації роботи та знаходження самого «user-friendly» дизайну, який буде до смаку якомога більшій кількості користувачів.

За будь-яку ціну розробники намагаються завоювати увагу користувачів.

Все вищезгадане знаходиться на поверхні самого застосунку, або ж на не досить глибокому рівні. Але якщо опускатися на глибокий рівень організації роботи мобільних застосунків, то тут справа стосується персональних мобільних пристроїв, які накопичують в собі велику кількість приватної інформації. Тому з появою різних типів застосунків розпочалася ера накопичення особистих даних. І саме це питання часто хвилює користувача, але не завжди цей процес можливо контролювати.

З цього моменту на поверхню виходить одна з головних проблем сьогодення для ринку мобільних продуктів – захищеність застосунків та безпека персональних даних їх користувачів.

Ця проблема починається зі звичайного використання персональних даних типу особистих вподобань, біометричних даних, приватних фото, що використовувались застосунком чи навіть скрите відслідковування геометричних даних, тобто місць пересування користувача. А на глобальному рівні знаходиться викрадення чи гірший випадок – продаж великих баз даних електронних поштових адрес з метою масової рекламної розсилки – «спаму».

Таким чином використання мобільних застосунків в житті людини може відігравати немалу роль, як з боку зручності та ультимативності програми, що полегшує життя, так я з боку збереження чи поширення приватних даних користувача. Від тих чи інших загроз користувач може вберегтись власними силами – забороняючи доступ до тих чи інших сегментів даних та мультимедіа, що знаходяться на мобільному пристрої. Але та частина, що вже лежить в реалізації застосунку, вже ніяк не може бути контрольована користувачем.

Для більш точного формулювання всіх аспектів загроз від користування мобільними пристроями необхідно проаналізувати перш за все структуру застосунків, поширені архітектури розробки, відомі способи злому та порушення цілісності таких інструментів взаємодії з персональними гаджетами. Також варто дослідити поширені сценарії використання тих чи інших мобільних застосунків, популярні методи легального використання приватних даних користувача за його згоди та інші варіанти порушень.

Для забезпечення необхідного рівня захисту застосунків необхідно детально дослідити всі вищезгадані фактори, розглянути популярні та ефективні способи протистояння загрозам та сформулювати технічну модель захисту мобільного застосунку, що буде базуватись на правилах, технічних рішеннях, обмеженнях та керівництвах з використання.

Актуальність роботи полягає у тому, що ринок мобільних застосунків дуже стрімко розвивається, і саме тому існує великий ризик втрати приватної інформації, розсекречення персональних даних без згоди на те користувачів, спекулювання такими даними задля приватних цілей, що може призвести до негативних наслідків.

Саме тому в індустрії мобільних розробок необхідно визначити загальні та конкретні методи організації захищеності мобільних застосунків, що буде представлено в даній роботі.

Метою роботи є аналіз індустрії мобільних застосунків, їх архітектури, можливих вразливостей в різних типах архітектури, загроз, що можуть порушити захищеність застосунків.

Завдання роботи полягає в тому, щоб проаналізувати ринок мобільних платформ та застосунків, обрати найпопулярніші платформи, проаналізувати їх загрози та засоби безпеки боротьби з ними, що реалізовані в цих системах, та на основі отриманих досліджень розробити архітектуру системи безпеки мобільного застосунку, що гарантуватиме стійку та захищену систему безпеки.

Об'єктом дослідження є сфера розробки мобільних застосунків.

Предметом дослідження є інструменти, правила та інструкції з забезпечення захищеності мобільних застосунків.

Методи дослідження полягають у детальному аналізі існуючих механізмів організації захисту мобільних застосунків на наявність вразливостей та методів запобіганню цих вразливостей.

Наукова новизна одержаних результатів полягає в тому, що запропоновано використання нового підходу до розробки системи безпеки мобільних застосунків

Практичне значення одержаних результатів полягає в тому, що розроблені у ході роботи методики та архітектури можуть бути застосовані на реальних проектах розробки мобільних застосунків, а також при модернізації вже існуючих систем безпеки.

1 МОБІЛЬНІ ЗАСТОСУНКИ ТА ЇХ АКТУАЛЬНІСТЬ

Мобільні застосунки - це програмні застосунки, призначені для роботи на смартфонах, планшетах та інших мобільних пристроях. Вони, як правило, доступні через магазини застосунків, якими керують власники мобільної операційної системи. За прогнозами, до 2020 року мобільні застосунки зароблять приблизно 189 мільярдів доларів США доходів через мобільні магазини та вбудовану рекламу. Серед найпопулярніших операційних систем - оригінальні магазини Apple Store App Store, Google Play, а також Windows Phone Store та BlackBerry App World. Станом на березень 2017 року в магазині Google Play було доступно 2,8 мільйона програм та 2,2 мільйони програм, доступних у магазині Apple App Store - двох провідних магазинах застосунків у світі.

Оскільки мобільні застосунки спочатку пропонувалися як інструменти для підвищення продуктивності та отримання інформації, на прикладі календаря, електронної пошти чи інформації про погоду, ринок швидко розширився завдяки потребам користувачів та наявності інструментів для розробників. Найпопулярнішою категорією застосунків серед користувачів Apple iOS є утиліти, серед яких інтеграції соціальних мереж, інструменти для перегляду та редагування фото та відео, ігри. Інструменти, засоби зв'язку, відеопрогравачі та різноманітні редактори, навігатори, місцеві телепрограми та газети – це все головні категорії застосунків для Android у світі. У Сполучених Штатах мобільні користувачі витрачають більшу частину свого цифрового часу (що залежить від їхньої вікової групи) з мобільними музичними програмами. Програми миттєвих повідомлень - це друга за популярністю категорія мобільних застосунків у США. Варто зазначити, що мобільні месенджери та соціальні мережі набули популярності не лише в Сполучених Штатах, але й у всьому світі. WhatsApp, що є провідним інструментом для мобільних повідомлень у світі, має 201 мільярда активних користувачів на 2017 рік.

Багато найпопулярніших об'єктів мультимедіа доступні не тільки через доступ у браузері, але існує також можливість їх використання у спеціалізованих

мобільних застосунках, розроблених компаніями-поставниками того чи іншого контенту. Популярні приклади – мобільні застосунки для соціальних мереж на чолі з Facebook, з яких понад 1,1 мільярда користувачів активно користуються мережею лише з мобільних пристроїв. Facebook – це застосунок для смартфонів із найвищим рівнем охоплення аудиторії в Сполучених Штатах, оскільки до програми соціальної мережі було доступно майже 80 відсотків мобільної аудиторії в країні. Facebook Messenger має другий найвищий рівень охоплення в країні, а YouTube займає третє місце. Для багатьох повсякденних дій американці використовують Facebook: щоб ділитися своїми новими інтересами, виражати свою творчість, розповідати про свій відпочинок чи планувати спільне проведення вихідних з друзями. Практично для кожної активності людини в цьому мобільному застосунку передбачено окремий модуль, за допомогою якого можна організувати захід, отримати порцію контенту чи зв'язатися з людиною.

В Україні досить схожа ситуація. Говорячи про месенджери в нашій країні безумовно лідирує Viber, яким як мінімум раз на місяць користується 94% власників смартфонів (близько 40% всього населення України). Facebook Messenger займає друге місце з величезним відривом, а WhatsApp – один з найменш поширених месенджерів. Інформація щодо використання застосунків на смартфоні на базі мобільної операційної системи Android в Україні за 2018 рік була надана компанією Admixer та базується на аналізі даних використання TNS, що відображено на Рисунку 1.1.

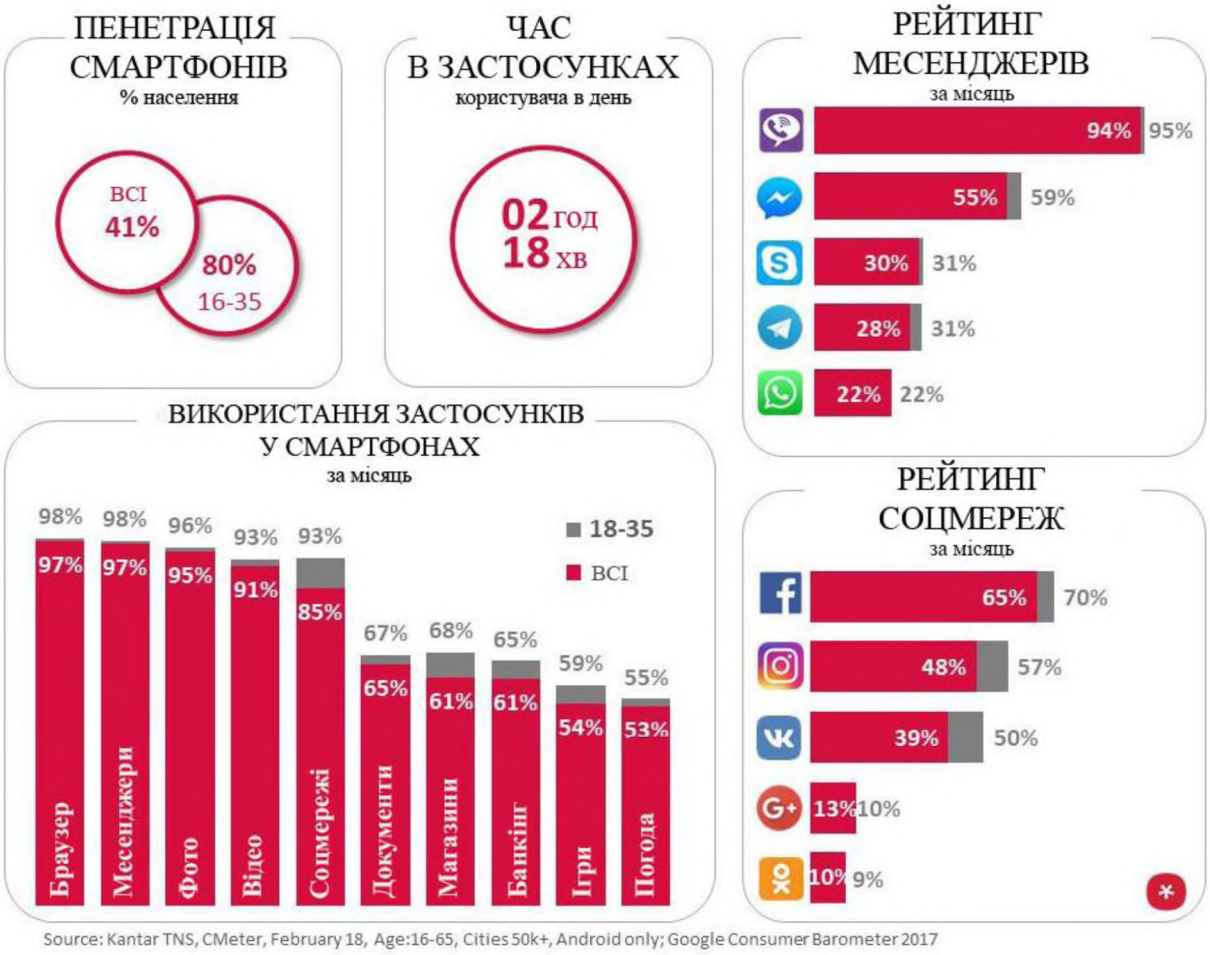

Рисунок 1.1 – Використання мобільних застосунків в Україні

В середньому користувач Viber проводить в додатку 30 хвилин в день. 45% цього часу складають дзвінки і відеодзвінки. Один користувач відправляє більше 20 повідомлень на день, а 35% користувачів користуються стікерами. Розмір аудиторії в Україні на сьогоднішній день компанія не розкриває.

WhatsApp охоплює тільки 22% користувачів смартфонів на місяць, і його в Україні вже випередив Telegram з охопленням 28%. Також в нашій країні все ще популярний Skype (30%).

Серед соцмереж після заборони російських сайтів в Україні впевнено лідирує Facebook з охопленням 65%. Колись лідируюча соцмережа «ВКонтакте» (39%) сьогодні поступилася місцем Instagram (48% охоплення) і посідає третє місце.

Дивно, що на четвертому - Google+ з 13% охоплення, адже цю компанія Google офіційно об’явила про закриття цієї мережі, а на п'ятому «Однокласники». Настільки низьку позицію другої російської соцмережі можна пояснити тим, що аудиторія "Однокласників" переважно більш зрілого віку і в основному заходить з комп'ютера, в дослідженні ж враховували саме додатки. Також варто додати, що доступ до російських соцмереж отримується незвичним шляхом – з використанням VPN підключень задля обходу блокування функціонування цих сервісів на території України відповідно до Указу Президента України Про рішення Ради національної безпеки і оборони України від 28 квітня 2017 року "Про застосування персональних спеціальних економічних та інших обмежувальних заходів (санкцій)".

Відповідно до статистики в 2017 році аудиторія месенджера Viber в Україні збільшилася на 50% в порівнянні з 2016 роком. Відомо, що на кінець 2016 року в сервісі було зареєстровано близько 16 млн акаунтів з України. У компанії відзначають, що зараз месенджер «є у кожного другого жителя України».

Таким чином ситуація з месенджерами доводить, що мобільна функція обміну повідомленнями в України досить поширена і нею користується більшість населення нашої країни.

1.1 Тенденції розвитку ринку мобільних застосунків

2017 рік був феноменальним для галузі розробки мобільних застосунків.

У промисловості з'явилася нова мова програмування, Kotlin, яка стала офіційною мовою від Google. В свою чергу компанія Apple з допомогою партнерських відносин посилила свою владу в галузі інтелектуальної власності. AR та VR технології отримали нову порцію підтримки від розробників «hardware», в тому числі від розробників з Купертино.

Також варто нагадати, що серед всіх гучних розробок у світ повномасштабно увійшло використання такої технології нашого часу як Blockchain.

Думка про те, що вищезазначені технології це межа розвитку ринку мобільних застосунків, досить хибна. Тенденції текучості та хаотичної зміни стану того чи іншого ринку в наш час – досить поширене явище. Так само можна сказати про ринок мобільних послуг, в якому за 5 днів анонсів та презентацій можна проголосити 2018 рік найцікавішим у світі для розвитку мобільного програмного забезпечення.

1.1.1 Мобільні платежі

Цього року очікується, що більша кількість користувачів звернеться до своїх мобільних телефонів, щоб робити щоденні платежі, оскільки люди, як правило, не користуються готівкою, а віддають переваги безготівковому розрахунку. Тому в 2018 спостерігається масове поширення пунктів прийому безготівкових платежів, що значно пришвидшує та додає зручності в здійсненні покупок.

До 2020 року ринок мобільних платежів зросте до 503 млрд. доларів, що становитиме 80% річного приросту від 2015 року до 2020 року.

Незважаючи на те, що вже існує безліч застосунків для здійснення мобільних платежів, що працюють по всьому світу, дохід від мобільних розрахунків збільшиться внаслідок того, що вартість річної трансакції, здійсненої через програми mPayment, є високою в США та світі. Очікуваний приріст доходу від мобільних платежів зображено на Рисунку 1.2.

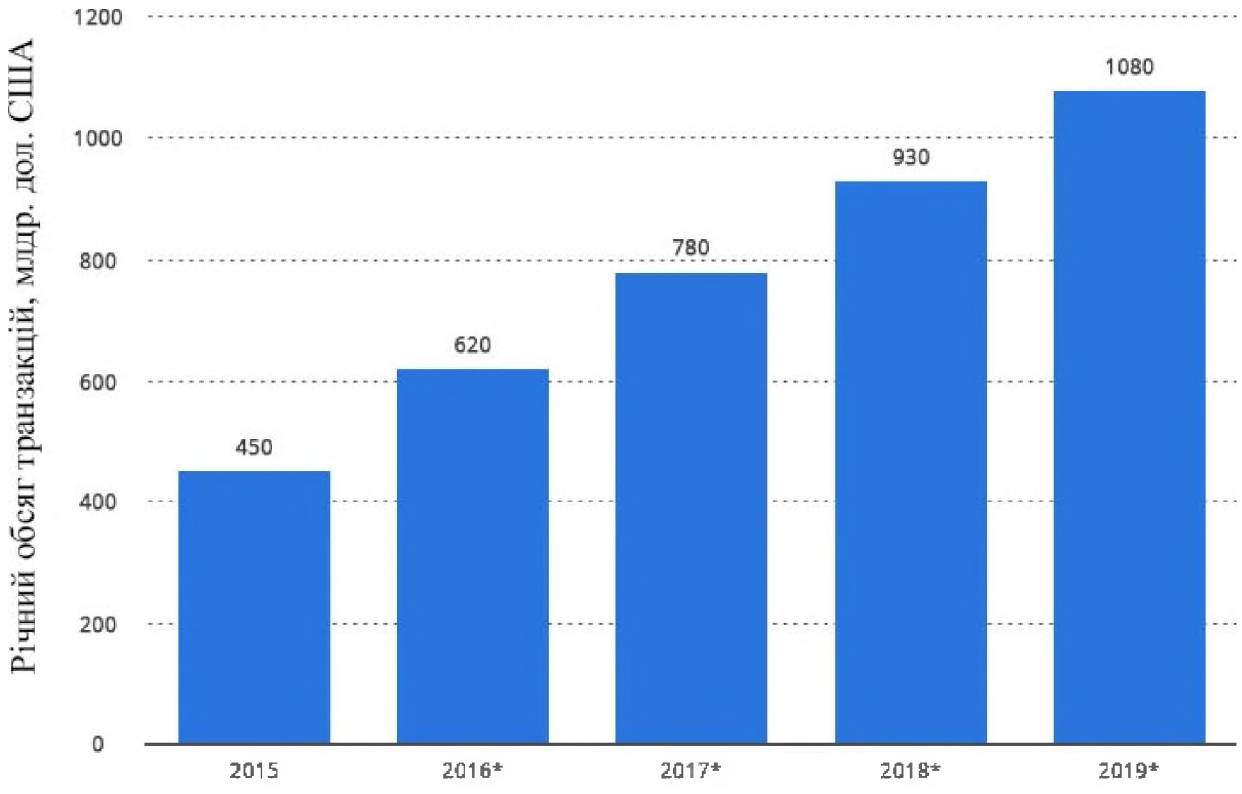

Рисунок 1.2 – Обсяг річних транзакцій у світі

1.1.2 Blockchain

9 з 10 урядових фірм США планують інвестувати в блокчейн для управління фінансовими операціями, управління активами, управління контрактами та регулюванню пропозицій комплаєнсу. Очікується, що одна третина банків прийме комерційний блокчейн у 2018 році у всьому світі.

У той час як біткойн – важливий елемент Blockchain, вже займає своє місце у світі технологій, 2018 року очікується та спостерігається вихід цієї технології за межі цифрового платежу. З кожним днем, коли технологія отримує все більше і більше інвестицій, галузі запроваджують використання блокчейн для багатьох інших цілей, ніж просто у випадку криптовалютних транзакцій.

Існують вже бренди, які почали пропонувати технологію, щоб обслуговувати свої смарт-контракти в галузі нерухомості, а також існують інші компанії, які використовують Blockchain Mobile Apps як додаток безпеки в системі управління ланцюгами менеджменту.

1.1.3 PWA

PWA – це вид застосунків, що не потрібно встановлювати в магазинах застосунків. Він поставляється в комплекті з різноманітними інноваційними функціями, такими як швидкі оновлення, швидша навігація, робота у автономному режимі, push-сповіщення тощо.

Фактори, такі як надійність, швидкість, взаємодія, доступність та економічність, разом з багатьма іншими, приносять прогресивні веб-застосунки у відповідність з їхніми першоджерелами – стаціонарними мобільними застосунками. Незважаючи на те, що корінні застосунки все ще мали перевагу в 2017 році, зміни, які плануються для PWA в 2018 році, змусять використовувати цей тип більшу кількість розробників.

1.1.4 IoT

Аналітики прогнозують, що IoT буде зростати з $ 157,05 млрд. в 2016 р. до $ 661,74 млрд. в 2021 р., і світ перебуває лише на початкових стадіях цього зростання.

Потреба користувачів щодо девайсів-одягу, яка виходить за межі задоволення їхніх медичних потреб, зростає. Потреба цього часу полягає в тому, щоб додати інновації в те, що можна зробити, натиснувши кнопку годинника або регулюючи власні окуляри. І це саме те, що спостерігається в 2018 році.

Кожна IoT та Wearable дискусія, що відбудеться чи вже відбулася в 2018 році, буде неправильною без згадки про ідею розумного будинку. У 2018 спостерігається розвиток адаптації таких технологій, як персональний віртуальний асистент Alexa, які допомагають користувачам керувати своїми розумними домашніми пристроями, такими як замки для дверей, камери, освітлення, розважальні системи та термостати тощо. Технологія, яка вже оцінюється на мільярд доларів, буде продовжувати зростати з часом, що зображено на Рисунку 1.3.

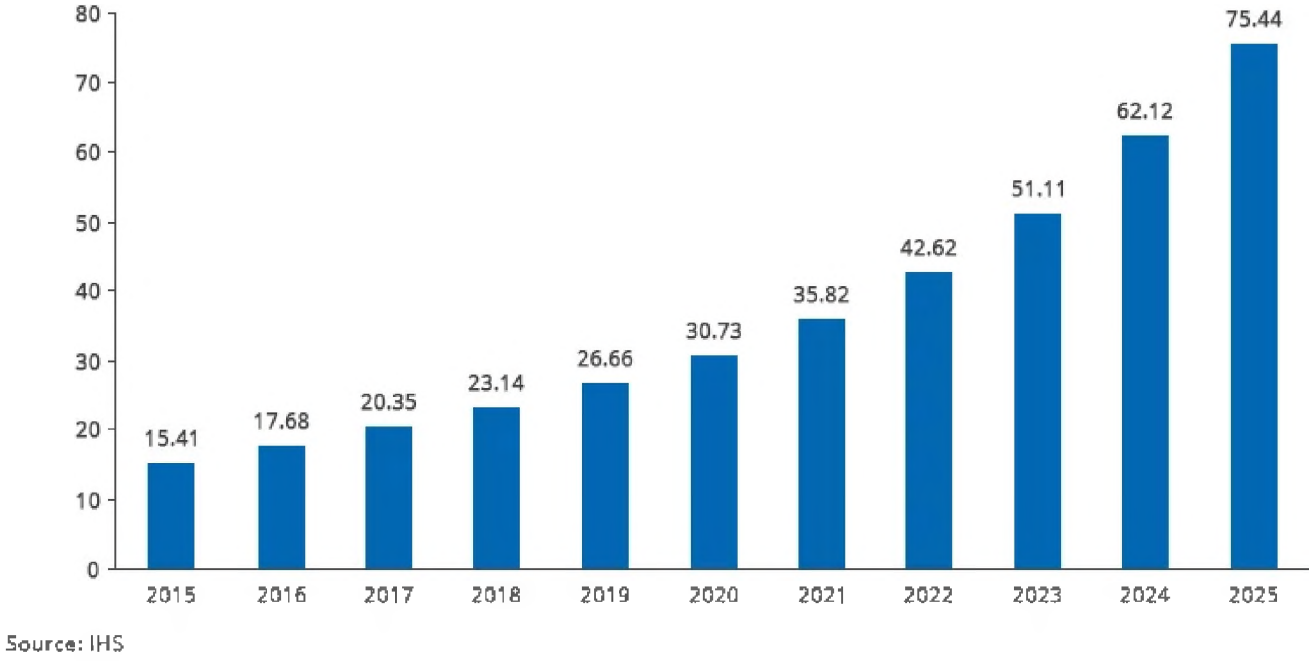

Рис. 1.3 Діаграма росту встановлених пристроїв IoT у світі, млрд шт.

1.1.5 AR та VR

Застосунки наступного покоління, зокрема ігри, напряму будуть стосуватися AR та VR. Технічні експерти передбачають, що до 2020 року виручка від продажу застосунків, які будуть використовувати ці технології становитиме близько 150 млрд. доларів.

Хоча дана технологія обмежувалася в основному ігровою та роздрібною сферою промисловості в 2017 році, 2018 рік показав нове бачення та використання AR/VR в галузях, де в порівнянні з класичними технологіями це було б дивно побачити. Мова йде про нерухомість, охорону здоров'я та освіту тощо. Відкриття, які AR / VR встановило для себе в 2017 році, значно покращилися до набагато більш глибокого рівня в 2018 році.

Складові доходів AR / VR не обмежуються лише AR Voice або AR Games та VR Games і VR Theme Parks, ринок з 120 мільярдами AR та 30 мільярдів доларів VR можна розділити на ряд інших продуктів, причому кожен з них зберігає свою значимість та вклад у визначенні майбутнього AR / VR у світі.

Впровадження AR / VR у мобільному застосунку цього у 2018 році не тільки вирішує проблему розкрутки застосунку у використанні, але також допоможе підвищити коефіцієнт доходності, якщо ваша програма має елементи електронної комерції.

1.1.6 Enterprise застосунки

За інформацією дослідження Adobe близько 77% власників бізнесу знайшли для себе користь від програми для підприємств та 66% збільшують свої інвестиції у цю нішу ПЗ. Також до 2021 року очікується 430 мільярдів доходів від ринків корпоративних прикладних програм.

Причини, чому корпоративні програми знайшли місце в цьому списку, пояснюється тим, що в сучасному корпоративному сценарії потреба налаштування робочого місця для людей стала справою минулого. За часів віртуальних працівників та віддаленого доступу до робочого місця потреба компаній, щоб постійного перебування їх співробітників на одному робочому місці нівелювалася.

Статистичні дослідження у вище зазначених питання приведено на Рисунку 1.4.

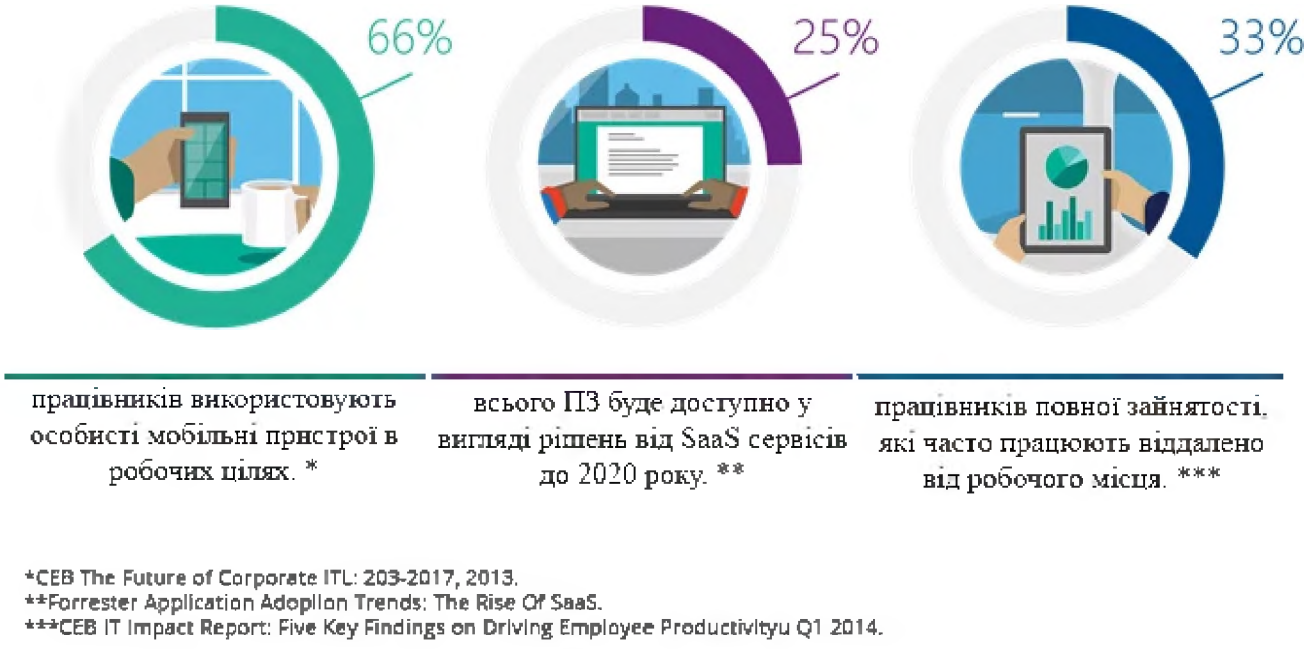

Рисунок 1.4 – Статистика віддаленої роботи та використання мобільних пристроїв у роботі

Починаючи з 2017 року підприємства поступово переходять з використання застосунків, що дотримуються їх внутрішніх стандартів галузі, до більш спеціалізованих застосунків, які мають виключно функціонал, що є необхідним для роботи того чи іншого підприємства, тим самим створюючи попит на більш різноманітні застосунки, що будуть задовольняти виключно їхні інтереси та цілі.

1.1.7 «On-demand» застосунки

У 2017 році користувачі витратили на 98% коштів більше на розважальні сервіси за запитом порівняно з 2016 роком. Сюди також відносяться користувачі, які бажали отримати інші послуги на замовлення, такі як стрижка, їжа, прання

речей та інше. Ця кількість витрат значно збільшилася у різноманітті та у обсязі в 2018 році.

Саме необхідність миттєво задовольняти кожну потребу за запитом призвела до збільшення попиту на дану технологію. У той час, як попит на послугу вже проник в ряд галузей, багато інших сфер побуту розцінюють дану технологію як початок власних розробок у найближчі часи – B2B активно розгортає кампанію використання економіки за запитом, щоб постійно перевіряти нову інформацію про співробітників чи отримувати результат їх роботи людини виключно за необхідності. Навіть сучасним галузям доведеться знайти нові інноваційні потреби об'єктів за запитом, щоб залишитися на вершині їхньої конкуренції.

1.1.8 Миттєві застосунки

Миттєві застосунки дуже зручні і це значною мірою залежить від залучення більшої кількості користувачів. Коли виникає необхідність встановлення застосунків, що як правило займають досить багато пам’яті, утворюється завада у вигляді низької пропускної можливості завантаження застосунків – бар'єр між застосунками та Інтернетом. Легкість, яку Android Instant Apps принесе в світ мобільних приладів з більшою адаптацією, значно покращить враження від використання мобільних застосунків в 2018 році.

Миттєві програми - це функція магазину Google Play, яка дозволяє використовувати застосунок без повного завантаження його на свій телефон. Все, що потрібно зробити, це знайти необхідну програму в магазині, а потім просто відкрити програму. Застосунки навіть дозволяють виконувати певну активність у застосунку, який не встановлений на вашому пристрої, лише одним натисканням на URL-адресу.

Забезпечуючи користувачам можливість користуватися застосунками, не прикладаючи жодних зусиль і не втрачаючи пам’яті телефону, це зробило Instant

Apps хітом у світі мобільних приладів. Доки лише кілька програм у магазину Google Play 2018 року побачили опцію "Спробуйте зараз". Очікувано, що з часом у більшій кількості застосунків з'явиться така опція, що збільшує шанси застосунку на популярність бренда.

1.1.9 «MHealth» застосунки

З понад 100 тисяч застосунків у галузі охорони здоров'я на ринку та прогнозу 1,7 мільярда завантажень програм mhealth на iOS та Android до 2017 року, є багато можливостей для більшої кількості застосунків з якіснішими систематиками та більш персоналізованими результатами.

Часи, коли додатки mHealth були пов'язані лише з показом кількості кроків, вжитих людиною, або для замовлення зустрічі зі своїм лікарем, вже давно пройшли. Сфера mHealth зараз повністю змінилася.

Цей рік вважається початком ери застосунків Електронних Медичних записів (EMR), які допомагають вести облік вашої історії хвороби, обробляти ваші медичні рахунки і вести та підтримувати перевірку на серйозні проблеми зі здоров'ям, такі як астма, діабет та рак по телефону.

Хоча в першу чергу використання такого типу рішень націлено на пацієнтів, лікарі також отримують власне поліпшення лікарського становища з використанням накопичених в застосунку даних, пов'язаних з прийняттям призначення та переглядом історії хвороби, для здійснення серйозних операцій за допомогою додатків на основі VR.

1.1.10Крос-платформні застосунки

Широке охоплення користувачів, поліпшення маркетингу, економічності та збільшення прибутків - це лише деякі з факторів, які підштовхнули світ розробки застосунків до використання програм для крос-платформ.

З огляду на те, що в 2017 році кількість користувачів, які працюють на Android і iOS, стрімко збільшується, розробники використовують стратегію більшого прибутку за менші затрати зусиль та схиляються до створення застосунків, що працюють на обох платформах. Існує ряд проблем, які вони вирішують шляхом інвестування в крос-платформні програми, такі як висока вартість, пов'язана з розробкою окремих застосунків для Android і iOS, використання різних продуктів на різних платформах та обмежене охоплення аудиторії серед інших операційних систем.

На даному етапі розвитку такого типу застосунків одними з головних проблем такого методу розробки є привітність користувацького інтерфейсу та швидкість роботи самого продукту

Щоб видалити ці вади, які поставляються з крос-платформними програмами, активно використовується фреймворк React Native.

Незважаючи на існування інших рішень, таких як NativeScript та Xamarin, React Native має однозначну перевагу в тому, що він є найкращим крос-платформним інструментом, що активно використовується при розробці мобільних застосунків. Переваги React Native полягають у наступному:

- Знання універсальної мови програмування JavaScript достатньо для розробки цілісного крос-платформного застосунку;
- React Native застосунки досить гнучкі при розробці та високопродуктивні;
- Застосунки повністю реалізовуються інструментами Native Controls;

- Така реалізація дозволяє розробникам повторно використовувати свої рішення.

1.1.11 Чат-боти та штучний інтелект

На даний момент тенденції користувачів вказують на те, що жорстка взаємодія з провайдером послуг є популярнішою, канали зв'язку, що відкриті цілодобово, більше привертають увагу користувачів, приховано пропонуючи стабільно постійну підтримку продукту онлайн. 52% споживачів вважають за краще взаємодіяти з підприємствами через застосунок для обміну повідомленнями, а не по телефону або особисто.

Хоча чат-боти зазвичай і далі виконують функцію відповідати на запити користувачів за попередньо запрограмованими шаблонами, але з використанням штучного інтелекту вони тепер мають можливість відповідати на запитання, які ще не були задані, не були попередньо продумані розробниками та жорстко вшиті в програмний код. А наступним кроком стає можливість розпізнавання суб'єктів, що знаходяться по ту сторону екрану, відповіді на питання чи це бот, чи це жива людина, що дихає.

Чат-боти вже знайшли своє місце в ряді галузей у всьому світі. Розподіл галузей використання чат-ботів зображено на Рисунку 1.5.

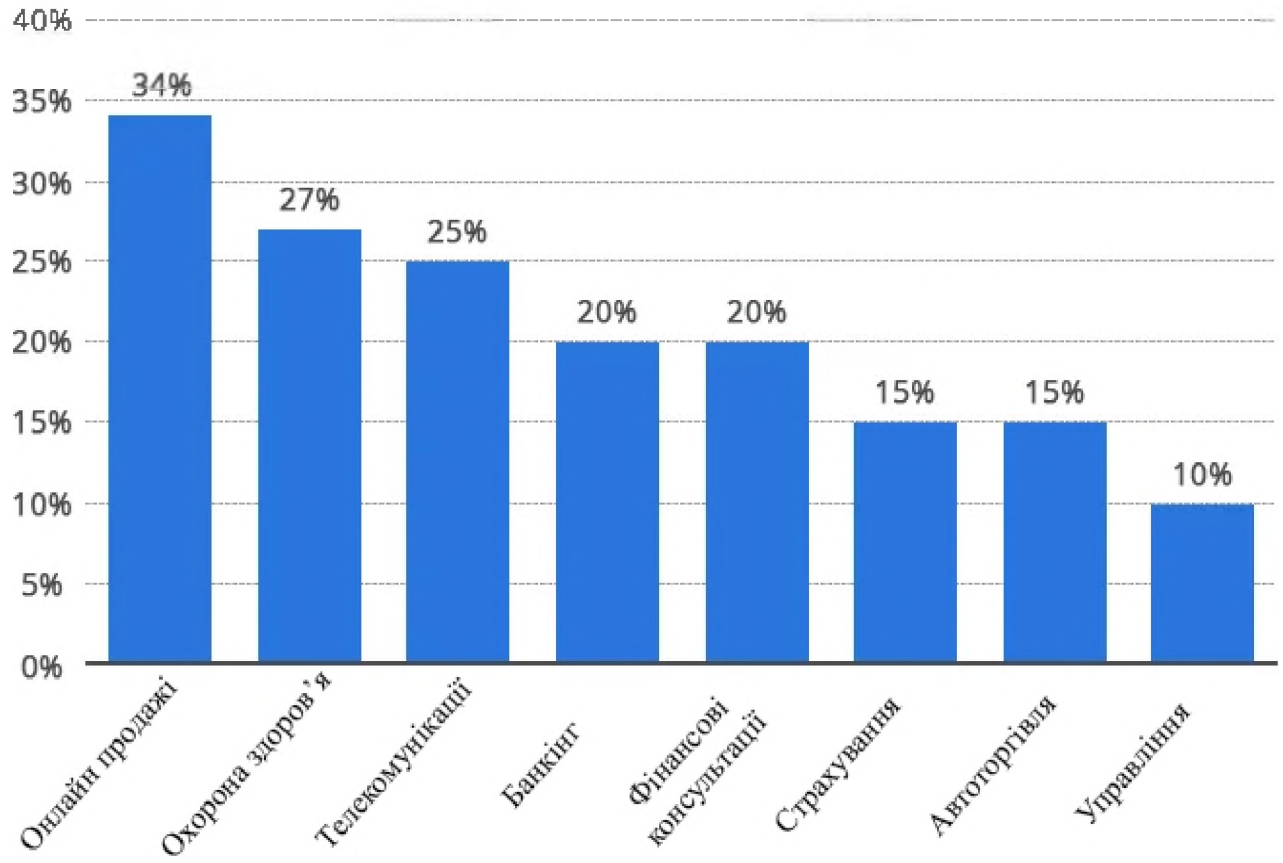

Рисунок 1.5 – Відношення використання чат-ботів в різних сферах

1.2 Аналіз розвитку мобільних платформ

За прогнозами, у поточному році загальний прибуток від магазинів мобільних додатків досягне $ 50,9 млрд., Згідно з новим звітом від програми Annie. Це означатиме зростання на 24% з валових доходів у розмірі 41,1 млрд. Доларів США з 2015 року. Аналіз передбачає, що до 2020 року дохід від магазину додатків перевищить $ 101 млрд. З огляду на 284 мільярди завантажень. Згідно з прогнозами Statista, в 2020 році валовий річний дохід перевищить $ 189 млрд. Незважаючи на те, що дані різних дослідників трохи відрізняються, загальний висновок такий: ринку далеко до насичення. Прогнози App Annie підтвердилися в звітах Forrester про те, що до кінця 2016 року лише 46% населення світу були власниками

смартфонів. Це свідчить про те, що широко обговорювана мобільна революція тільки починається.

Незважаючи на те, що ігри, ймовірно, залишатимуться великим рушієм зростання, у звіті також вказувалося на можливе збільшення доходів від негромадських додатків. У 2015 році дохід від не ігрових рішень становив 6,3 мільярда доларів, і очікується, що він збільшиться до 9,4 мільярдів доларів у 2016 році та 26,4 мільярда доларів у 2020 році. "Зокрема, потоки музики, потокові відео та додатки для знайомств стали головними факторами доходу на цих ринках та ми сподіваємось, що вони продовжують будувати свій успіх", – сказано в повідомленні.

Час, витрачений на засоби масової інформації та відеопристрої, збільшився на 93% протягом 2014-2015 років лише для телефонів на базі Android. Якщо ріст поточної тенденції до мобільного прослуховування продовжиться, особливо серед молодшої аудиторії, то цей показник, найімовірніше, продовжить зростати у найближчі роки. Програма Anni Retrospective 2015 також показала, що прибуток від музичних застосунків зростає у порівнянні з попереднім роком.

Інша статистика Forrester демонструє величезний розрив між провідними компаніями, для яких мобільні пристрої стали каталізатором перетворення їх бізнесу, і компаніями, які відносяться до мобільних пристроїв просто як до ще одного напрямку для розвитку. На початку 2016 роки тільки 18% опитаних компаній ставилися до першої категорії. Цей показник, як очікується, подолає 25% вже в наступному році.

Споживачі еволюціонують ще швидше, ніж бізнес. Сьогодні мобільний інтернет для багатьох користувачів став нагальною потребою.

Що стосується популярності, на передній план, ймовірно, вийдуть програми-агрегатори. Це інструменти, що дозволяють отримати контент з різних онлайн-ресурсів та об'єднати його в простому зрозумілому інтерфейсі. Контент може бути найрізноманітнішим: від останніх новин до популярних інтересів. Агрегатори призначені для тих, у кого немає часу або бажання відвідувати багато сайтів або

встановлювати багато програм. Серед популярних застосунків-агрегаторів - Flipboard, News360, Feedly і IFTTT.

Застосунки-агрегатори, як правило, завойовують популярність серед користувачів в тих випадках, коли вони зручні або покращують процес покупок. Наприклад, Facebook зробив це з додатком Messenger, яке дозволяє користувачам читати свої стрічки і викликати Uber.

Два гіганта мобільного розробки Android і iOS домінують на глобальному ринку смартфонів. Згідно з дослідженням Gartner, 87,8% проданих в третьому кварталі 2016 року смартфонів - частка Android. Цей показник на 3,1% вище, ніж рік тому. Частка ринку iOS - 11,5% (на 2% нижче, ніж в 2015). І хоча цей показник дуже малий для вже охопленої величезної частки ринку, це зростання дуже послаблює позиції інших гравців ринку. Windows, на чиєму рахунку 0,4% від всіх проданих смартфонів, в гонці мобільних платформ фінішувала третьою зі зниженням своєї частки на 2,5% за рік.

У Apple і Google - найбільші і популярні магазини застосунків. На сьогоднішній день можливість інших учасників рину вирватись перед цими гігантами здається досить хмарною та уявною, як за асортиментом мобільних застосунків, так і за підтримкою розробників ПЗ зі сторони перспективності використання інших платформ окрім Google та Apple.

За оцінками InMobi, 55% розробників заробляють менше $ 1000. Більш того, третина розробників додатків у всьому світі не змогла домогтися 10 000 завантажень своїх продуктів. Нерівність доходів яскравіше виражене для розробників на Android, тоді як серед iOS-розробників розподіл доходів більш збалансовано.

З 2016 року понад 25% iOS-розробників генерували більше $ 5 000 місячного доходу. Серед Android-розробників такого показники досягли тільки 16%.

Дивні дані можна побачити аналізуючи статистику прибутковості за місяць саме платформи для розробника. За оцінками популярного журналу Forbes, платформа iOS заробляла в середньому $ 4 000 в місяць, на другому місці був

Android з доходом близько $ 1 125, а на третьому - аутсайдер Windows Phone і всього $ 625.

Однак ця ситуація кардинально змінилася в 2016 році. Згідно з даними Statista, додаток Windows Phone почав приносити в середньому $ 11 400 в місяць, тоді як додаток iOS генерує $ 8 100, а Android - $ 4 900. При цьому 75% розробників є прихильниками Android. У них в планах збільшувати свій дохід, роблячи продукти під Android. Така тенденція скоріше за все завдячує активному просуванню Microsoft власного виробництва, що побудовані на базі ОС Windows Phone, яка популяризувала ідею повної синхронізації власного телефону з персональним комп'ютером.

Останні кілька років спостерігається безпрецедентне число людей, що поспішають розробляти мобільні додатки для iOS та Android. Але, дивлячись на отриману користувальницьку базу по кожній з платформ та інформацію про виплати, зроблені різними компаніями, здається, що переважна більшість розробників залишається з відносно невеликим доходом.

Ситуація в цій галузі полягає в тому, що Google домінує на ринку мобільного зв'язку з 2 мільярдами щомісячно активних девайсів, тоді як Apple посідає друге місце з 2 мільярдами придбаних пристроїв iOS, а Microsoft посідає третє місце з приблизно 12 мільйонами проданих телефонів Windows (переважна більшість з них - 81% , продавалися під брендом Nokia NOK -0,72%).

За допомогою різних форумів, спрямованих на залучення розробників, кожна компанія обробляє розмір своїх ринків в різних варіантах.

Компанія Apple на конференції розробників WorldWide розповіла про 1,25 мільйони застосунків у магазині App Store, що генерують близько 50 мільярдів завантажень і 5 мільярдів доларів доходів розробникам за останній рік. Для компанії це є ознакою гордості, що вони оплачують розробникам їх труди таким впевненою сумою грошей. Внутрішні дані з магазину застосунків, зібрані з джерел, близьких до компанії, вказують на те, що цифри відповідають фактичним платежам, які виставляються розробникам.

У Google I/O, найбільшої конференції розробників Android, Google заявляв про 150 000 розробників, що відповідають за розробку більш ніж 800 000 додатків. Незважаючи на те, що компанія не розбиває кількість доходів на своїх додатках, останні дані, що містяться у фінансових звітах, здавалися десь приблизно 900 мільярдами доларів для розробників "протягом останніх 12 місяців" за 2013 рік, а обговорення з зовнішніми аналітиками дослідження поставили кількість завантажених додатки з магазину Google Play складають близько 48 мільярдів доларів, близьких до заявлених Apple.

Відповідно сьогодення динаміка росту популярності мобільного середовища вражає. Улітку 2018 року екосистема мобільних додатків – одна з найбільших галузей на цій планеті – перетворилася на 10 таких екосистем. Вона охоплює мільйони розробників додатків – буквально мільярди власників смартфонів, які щоденно користуються мобільними програмами, та компанії, які керують цією екосистемою – Apple, Google і, меншою мірою, Amazon і Microsoft.

Apple Store App Store побудований за керованою моделлю і це означає, що Apple контролює якість застосунків та їх відповідність певним стандартам, визначеним Apple. Позитивною стороною цієї моделі є можливість надавати користувачам застосунки без шкідливих програм, помилок та вмісту, що є неприйнятним для неповнолітніх. З негативного боку існують неминучі проблеми з тим, що помилково відхиляються публікації, які не несуть шкоди, та відповідно з похибкою були оцінені як шкідливі. Причиною для компанії Apple для вибору моделі такої керування є те, що надзвичайно важливо, щоб компанія зберігала якість мобільних застосунків з урахуванням високої апаратної стандартності та забезпечувала максимально високий рівень задоволеності клієнтів, адже за ідеологією Apple саме користувач та його досвід експлуатації виробів компанії знаходиться на першому місці.

Процес публікації застосунку Google Play для магазину Google передбачає набагато менш суворі правила для розробників під систему Android, які відповідають вимогам. Процес перегляду застосунків займає набагато менше часу, ніж Apple App Store. Як наслідок, розробники застосунків можуть публікувати свої

програми набагато швидше, але з іншого боку, значна частина зловмисних застосунків Android публікується в Google Play, і це постійна проблема, з якою Google має справу. Google бачить свій магазин застосунків Android ще одним каналом для розповсюдження програмного забезпечення, де компанія може розміщувати реклами та отримувати прибуток від неї.

З точки зору бізнесу, для Amazon його Amazon AppStore служить пропозицією додаткової зручності для людей, які довіряють бренду компанії та купують у інтернет-магазині Amazon велику кількість різних продуктів. Amazon не розглядає свій магазин застосунків з точки зору отримання прибутку. Саме тому Amazon надає своїм клієнтам додатки для Android і не інвестує в створення власної екосистеми мобільного програмного забезпечення, як це робить Apple та Google за допомогою iOS та ОС Android відповідно.

Тим часом, Microsoft задекларувала близько 160 000 застосунків у своєму магазині, що були подані від 45 000 розробників. У недавньому інтерв'ю представники Microsoft стверджували, що середньостатистичний користувач завантажив 54 застосунки, що дозволило встановити на сьогоднішній день загальну кількість скачувань близько 650 мільйонів. Незважаючи на те, що компанія не розбиває дані для свого мобільного підрозділу, розглядаючи варіації лінії кредиторської заборгованості своїх 10-Q за останні кілька кварталів до і після того, як вони представили свій магазин додатків показує варіацію до $ 100 млн. З 2011 року що можна віднести до магазину додатків.

Загальна статистика відображена на Таблиці 1.1.

Таблиця 1.1 – Статистика найпопулярніших платформ мобільних застосунків станом на 2017-2018 роки

	Google	Apple	Microsoft
К-ть користувачів(млн.)	900	600	12
К-ть застосунків (тис.)	800	1250	160
Можно)К-ть розробників (тис.)	150	235	45

Продовження Таблиці 1.1

К-ть завантажень (млрд.)	48	50	65
Заробіток розробників ($ млн.)	900	5000	100

Виходячи з даних таблиці можна зробити висновок, що Apple перемагає в цій гонці продаж. Але питання заробітку саме розробників – потенційних популяризаторів платформи, залишається нерозкритим.

В Таблиці 1.2 зображено середній дохід для розробників за їх продукти.

Таблиця 1.2 – Середня статистика доходу розробників від мобільних застосунків

	Google	Apple	Microsoft
Застосунків на розробника	5	5	3
Скачування застосунку	60 000	40 000	4 062
Дохід від одного скачування	$ 0.01875	$ 0.1	$ 0.1538

Виходячи з цих даних, середньостатистичний розробник на цих платформах досить зайнятий, розробляючи від 3 до 5 застосунки залежно від платформи. Цікаво, що Android виходить переможцем у кількості завантажень, але це значною мірою нівелюється суттєвою різницею доходів інших конкурентів, а середня вартість одного завантаження застосунку оцінюється в приблизно 2 центи для розробника; розробник для платформи Apple в той же час отримує в 5 разів більше доходу, залучаючи 10 центів з кожного з 40 000 потенційних завантажень власного застосунку. Але цікава річ виходить з даних, що платформа Microsoft виявляється набагато більш корисною для своїх розробників, в результаті чого дохід за завантаження досягає аж 15 центів (факт, який компенсується кількістю завантажень, що являють собою лише 10 відсотків від того, що можуть запропонувати інші платформи).

З середньою вартістю платної програми на суму $ 0.99 спостерігається прямий вплив безкоштовних програм на ці ринки. Величезне лідерство Android у наданні безкоштовних програм сильно знижує середній дохід, який виплачується розробникам, тоді як менша кількість вільних програм на платформі Windows може допомогти отримати більше прибутку.

Максимізація розвитку мобільних обчислювальних рішень привела до нарощування потужності у відносно малих за розмірах пристроях. Так на світ з'явилися високопродуктивні мобільні процесори, підтримка потужної графіки, якісні екрани і швидке інтернет-з'єднання. Ці покращення в мобільній індустрії перетворили смартфони в ігрові пристрої з обчислювальними можливостями ноутбуків. Відповідно до звітів App Annie, мобільні ігри, на які в 2011 році припадало менше 50% доходу від усіх мобільних застосунків, згенерували 85% доходів ринку мобільних застосунків в 2015-му. Це $ 34,8 млрд для всього світу.

Різко збільшився час, який користувачі проводять в різного виду застосунках. При цьому неігрові застосунки обійшли ігри. За даними Flurry Analytics, до кінця 2015 року ринок мобільних застосунків встановив новий рекорд по використанню застосунків:

- Застосунки для персоналізації пристроїв (оболонки, іконки, фонові зображення для основного екрану і екрану блокування) очолили список з приголомшливими 332% зростання використання за час користувацької сесії;
- Мобільні версії газет і журналів посіли друге місце з величезним зростанням в 135%;
- Інструменти підвищення продуктивності - на третьому місці: 125% зростання;
- Рішення для шопінгу і стилю життя показали 81% зростання і четверте місце;
- Додатки для подорожей, спорту, здоров'я та фітнесу - поряд з мессенджерами і соціальними програмами - зросли на 53-54%;

- Ігри виявилися єдиним аутсайдером: в них користувачі провели на 1% часу менше.

Таким чином роблячи висновки з тенденцій розвитку людства можна сказати, що суспільство поступово переходить в мобільний світ діджиталізації. Паперові газети заміняються електронними, походи в магазини за речами – переглядом одягу в інтернеті та подальшим його замовленням з доставкою, планування часу в рукописних щоденниках – електронними аналогами, що синхронізуються з усіма гаджетами користувача через хмарні сховища.

1.3 Варіанти монетизації контенту мобільних застосунків

Розробники застосунків створюють програми з різними цілями. Для одних головною метою є підвищити лояльність аудиторії до компанії та підвищити рейтинги на світових ринках, інші прагнуть залучити більше клієнтів, а деякі бажають отримати дохід від саме цього нового продукту. В частих випадках мобільні застосунки використовуються як додатковий канал маркетингу для бренду, який відрізняється від інших каналів тим, що починає працювати відразу після запуску.

Для того, щоб новостворений застосунок ефективно розпочав приносити заробіток його власнику, необхідно обрати вірну модель монетизації контенту для власного продукту.

На сьогодні існує декілька найпопулярніших моделей монетизації мобільних застосунків – дієвого засобу отримання «живих» грошей від роботи власного застосунку.

Безкоштовні застосунки з вмістом реклами

Це один з найбільш поширених способів отримувати дохід від своєї програми. Ніяких обмежень по завантаженню немає. Мета розробника – набрати

якомога більше користувачів. Дані про їх поведінку потім аналізуються і надаються рекламодавцям, готовим платити за розміщення реклами.

Мобільний застосунок Facebook – ідеальний приклад використання такої моделі. Його користувачі не платять соціальной мережі нічого, але Facebook збирає про своїх користувачів величезні масиви даних, запускаючи потім показ цільової реклами. У їхньому випадку ця модель роботи надзвичайно ефективна – соціальна мережа недавно заявила про 151% зростання доходів від реклами в другому кварталі 2015 року.

Переваги такої моделі полягають у тому, що застосунок може швидко набирати велику клієнтську аудиторію, яку привабила безкоштовність продукту. А це в свою чергу позитивно відобразиться на ринку мобільної реклами, яка вже перевершила обсяг ринку традиційної реклами на радіо, в журналах, газетах. Популярні застосунки досить швидко формують поведінкові патерни користувачів, що може знадобитися рекламодавцям, а це вже робить модель дуже ефективною при використовуванні цільової реклами.

Мінуси полягають у тому, що така модель є досить відомою і користувачів вже дратує реклама в застосунках, що в свою чергу призводить до відтоку користувачів. Також мобільна реклама в першу чергу обмежується малою площею розміщення на мобільних екранах. Ну а для нішевих продуктів, що створені для допомоги користувачу в обмеженій кількості інструментів, реклама буде виглядати досить неприродньо.

Freemium

При використанні Freemium (комбінація «free», з англ. – «безкоштовно», і «premium») основні можливості доступні користувачам безкоштовно, а розширена функціональність – за одноразову плату або по платній підписці. Зазвичай кількість тих, хто готовий платити, відносно невелика. Тому програми, які використовують цю модель, зосереджені на тому, щоб забезпечити максимальну кількість завантажень.

Незважаючи на серйозну критику її потенційно спекулятивних механізмів, модель Freemium чудово працює при продуманому застосуванні. Найкращий

приклад – мобільна гра Clash of Clans. Також чудовим прикладом зараз є гра Angry Birds. Компанія Rovio надає застосунок безкоштовно, але деякі функції приховані до моменту оплати, включаючи додаткові рівні, розширення можливостей птахів та інші. Гра дуже цікава, тому користувачі грають у неї місяцями і деякі купують додаткові можливості вже за гроші виключно через витрачений на гру час.

Але якщо запропонувати замало базового безкоштовного функціоналу, то користувачі перестануть користуватися застосунком, а якщо занадто багато, то вже не стануть купувати додатковий контент.

Платні застосунки

Така модель досить популярна серед розробників, що передбачає надання доступу до застосунку за одноразову плату. Вартість варіюється від 1 до 1000 доларів. Ключем до успіху застосунку може стати тільки приваблива презентація застосунку розробником, що я правило викладається на сторінці застосунку, де описуються всі ключові переваги продукту, від якого «не можливо відмовитись».

Переваги такої моделі полягають в моментальному доході відразу після покупки, а користувачі, придбавши ваш застосунок, скоріше за все будуть використовувати ваш продукт. Для користувачів головною перевагою буде відсутність реклами.

Серед мінусів виділяється висока конкуренція на ринку серед платних застосунків.

Внутрішні покупки

Такого типу застосунки передбачають продаж віртуальних чи реальних речей, що потрібні користувачам. Продаються такі речі, як віртуальні персонажі, одяг та спорядження для них, внутрішня ігрова валюта та інше.

Перевагами такого підходу є те, що система заробітку стає досить гнучкою, також внутрішні продажі мають мінімальний ризик, а можливість їх здійснення підвищує рівень лояльності користувачів.

З недоліків варто виділити те, що частку доходів з віртуальних продаж забирають каталоги застосунків.

Підписка

Дана модель монетизації схожа на Freemium. Але суть підписки полягає в тому, що за періодичну плату користувач отримує додаткового контенту, а не розширених функцій застосунку. Прикладом такого є обмежена кількість безкоштовних прослуховувань підкастів-новин застосунку Umano, в той час як необмежену кількість прослуховувань можна отримати тільки через підписку.

Перевагами є те, що користувачі, що використовують підписку, є досить лояльними, також стабільний дохід від періодичних надходжень мотивує розробників підтримувати актуальність контенту.

Але з іншої сторони архітектурно досить складно правильно застосувати модель підписки, адже бувають випадку, коли запропонований варіант не зовсім привабливий та дієвий.

Спонсорство

Серед інших ця модель виділяється своїм підходом – рекламодавець винагороджує користувача за виконання певних операцій в застосунку. Тут головним інструментом монетизації є бренди та рекламні агентства, які виплачують розробнику процент від нагород користувача.

Першою і напевно головною перевагою такої моделі є висока лояльність користувачів, які отримують бонуси. Також такий спосіб монетизації є досить гнучким у застосуванні.

Але в той же час використання такої нової моделі потребує наявності прямого контакту зі спонсорами та їх постійний контроль з боку розробника, адже саме спонсор і буде його головним генератором доходів.

Висновки до розділу 1

В даному розділі було проаналізовано актуальність мобільних застосунків на сьогоднішній час, було наведено статистичні та теоретичні дані, які доказують

привабливість використання мобільних застосунків як користувачами задля підвищення власної ефективності та якості життя, так і розробниками, як потужний метод генерування доходів.

2 АНАЛІЗ БЕЗПЕКИ МОБІЛЬНИХ ЗАСТОСУНКІВ НА БАЗІ ПРОВІДНИХ ПЛАТФОРМ

Виходячи з проведеного аналізу мобільного ринку доцільно буде описувати дві найбільші платформи застосунків – iOS та Android. Розробники зазвичай не оминають розробку окремих застосунків під кожну з ОС та часто на сайтах продуктів посилання на завантаження застосунку для обох операційних систем знаходяться поряд.

Для Anrdoid OS реалізована велика кількість застосунків, система є досить гнучкою в налаштуванні та також є контрольованою. Варто додати, що базовий захист таких систем в останніх версіях ОС знаходиться на високому рівні, тому пропонується увазі огляд системи забезпечення безпеки Android, що відразу присутня в ній.

В свою чергу iOS є свого роду обмеженою екосистемою. В налаштуванні це досить обмежена система, яка походить від спрощення Mac OS X. Але останні нововведення приносять в цю систему різноманітні покращення та більше інструментів для персоналізації використання пристрою.

2.1 Android OS

Android-пристрої набули популярності через свою відкритість для зовнішніх розробників та Linux подібну ОС, але в ній існують критичні відмінності, які зроблені для зручності використання та обслуговування мобільних пристроїв.

Структура ОС Android має чотирьохрівневу структуру, що зображена на рис. 2.1.

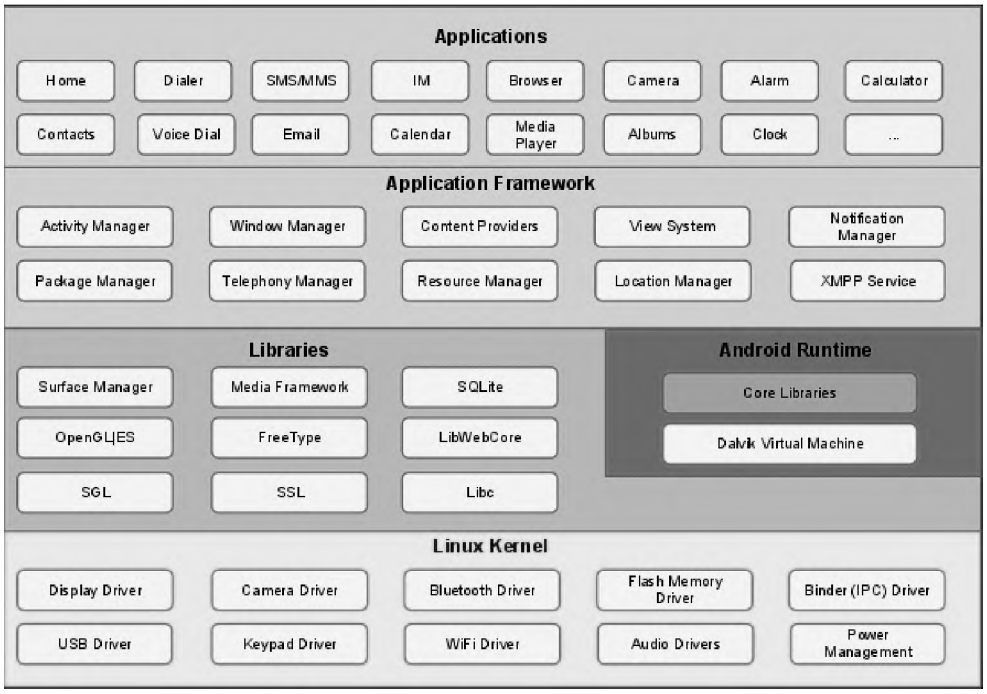

Рисунок 2.1 – Структура ОС Android

2.1.1 Ядро Linux Kernel

Як видно з рис. 1.1, в основі системи лежить ядро Linux Kernel, що виконує більшу частину роботи: контроль процесів, їх коректне виконання, дотримання прав доступу та використовується як hardware abstraction layer(HAL). Такий варіант організації системи був викликаний через те, що по функціоналу ядро Linux задовольняло всі необхідні критерії та є відкритою для налаштувань.

Відмінністю ядра Android від ядра Linux є саме HAL. Як відомо, існує два варіанти розміщення драйверів в системі Linux – вони або вбудовані в ядро, або розроблені у вигляді завантажуваних модулів. Оскільки недоцільно було б вбудовувати велику кількість драйверів в ядро мобільної системи, було розроблено HAL – проміжний рівень взаємодії між ядром та драйверами, що являє собою набір інтерфейсів, реалізація яких прописана в драйверах.

Ще одною відмінністю мобільного ядра є вбудовані деякі системи, що характерні тільки для Android. Серед них самим вагомими є Binder (модуль

забезпечення міжпроцесової взаємодії IPC/RPC), Asynchronous Shared Memory — Ashmem (драйвер роздільної пам'яті), Wakelocks (механізм, що передбачає затемнення екрану та відключення процесору), Low Memory Killer, Alarm та Logger.

Головний аспект безпеки цієї ОС реалізується за допомогою розмежування прав доступу, що реалізує саме ядро системи. Жоден застосунок не може отримати доступу до іншого застосунку без дозволу останнього. Такий доступ забезпечується за допомогою того, що кожен застосунок в такій системі наділяється власним ідентифікатором користувача UID та групи GID(як процеси в Linux) та також каталогом всередині каталогу */data*, що таким чином захищає дані застосунку за допомогою простих прав доступу на дозвіл зчитування застосунком власних файлів та заборону робити це іншим. Також на рівні ядра унікальні UID та GID кожного застосунку використовуються для розподілу доступу до ресурсів системи – пам'яті та процесорного часу.

Описаний принцип роботи застосунків в Android називається пісочницею (sandboxing) – варіант розмежування доступу між застосунками, який локалізує та захищає персональні дані всіх застосунку системи та виключає можливість злоякісного програмного забезпечення викрасти цю персональну інформацію. В пісочниці знаходяться всі застосунки системи: такі що були попередньо встановлені та ті, що встановлювалися в ході експлуатації пристрою.

Не підпадає під цю категорію контролю тільки процеси, що працюють з правами root, а таких досить небагато: початковий процес zygote, що контролює виконання застосунків, та також деякі системні сервіси. Дякуючи такому обмеженню злоякісне ПЗ має доступ тільки до інформації, що зберігається на SD-карті, і тільки за умови, що користувач дав зводу на цей доступ.

2.1.2 Native user space рівень

До системної інсталяції Android також доступ закритий, адже всі дані інсталяції та автозапуску знаходяться на окремому блоці пам'яті NAND, що віртуально підключений до каталогу */system*. Доступ до цього каталогу за замовчування змонтований тільки на зчитування і не несе в собі ніякої конфіденційної інформації, а інформація, що циркулює в цьому каталозі також входить до пісочниці. Таким чином вмонтувати в автозавантаження шкідливий код чи модифікувати його просто так не вдасться. Злом такої безпеки можливий лише за використання root exploit – таким рішенням, що дозволяє наділити шкідливий процес правами суперкористувача.

Важливим фактором захисту на цьому рівні системи є жорстко запрограмовані значення деяких користувачів системи. Оскільки Android розрахувався як мобільна система, у якої зазвичай один користувач, то було прийнято рішення використовувати різних Linux users для забезпечення безпеки – для кожного застосунку власний користувач з власним UID, як було зазначено вище. Наступним рішенням було жорстко запрограмувати декілька стандартних user(користувачів) та ідентифікаторів в систему. Йдеться мова про *root* з ідентифікатором 0 та *system* з ідентифікатором 1000.

У випадку привілейованих програм(наприклад su) приймається обмеження на застосунки, які можуть визивати ці програми, адже якщо такий виклик здійсниться, то буде реалізовано отримання прав суперкористувача. В системі така проблема контролюється через перевірку UID процесу – спочатку програма перевіряє від чийого імені викликаний процес

2.1.3 Application framework рівень

Головною рисою розглядаємого рівня є взаємодія застосунків між собою. Програми повинні мати можливість отримати дозвіл на інформацію від системних процесів. Але оскільки застосунки та системні сервіси виконуються в різних процесах, то операційній системі потрібно реалізовувати механізм «спілкування» між процесами. Це стосується також і звичайних застосунків та їх взаємодії.

В Android системі таку важливу функцію міжпроцесної взаємодії (Inter-Process Communication) виконує Binder IPC фреймворк. Він дозволяє синхронно та асинхронно викликати методи віддалених об'єктів, забезпечує обмін дескрипторами між процесами.

Робота Binder організована за класичною схемою клієнт-сервер. Клієнт ініціює з'єднання та очікує на відповідь від сервера. На рис. 1.2 зображено простий приклад такого виклику, коли клієнт через proxy звертається до Binder, який знаходить необхідний сервіс та надсилає відповідь клієнту.

Також варто зазначити, що у випадку асинхронного виклику сервер відразу надсилає пусту відповідь клієнту.

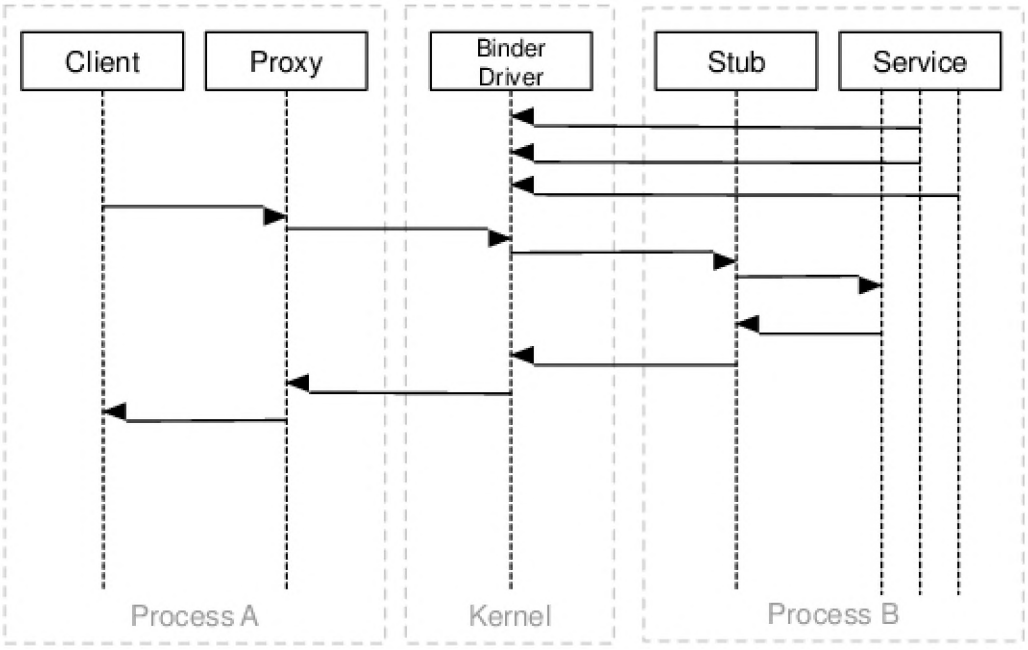

Рисунок 2.2 – Схема роботи фреймворку Binder

Вищерозглянуту взаємодію в рамках Binder забезпечує Linux device driver (драйвер пристрою), що знаходиться в каталозі */dev/binder*. Адресація між процесами відбувається за допомогою токенізації. Binder призначає кожному сервісу унікальний серед всіх процесів токен, який є фактично вказівником на цей сервіс, і якщо у клієнта є цей вказівник, то тільки тоді можливе використання процесом такого сервісу. Але спочатку клієнту необхідно отримати цей унікальний токен.

Процедура отримання токену виконується через Binder context manager, що у Android системі має назву Service Manager. Цей спеціальний сервіс за замовчування наділений токеном, значення якого відомо всім сервісам та дорівнює 0. Service Manager виконує функцію довідника, в якому зареєстровано набір сервісів, та після запиту клієнта на отримання сервісу з певним ім’ям менеджер надсилає у відповідь токен, що в подальшому клієнт зможе використовувати для взаємодії з цим сервісом. Відповідно перед використанням певного сервісу його необхідно

зареєструвати в менеджері, що є однією з ланок безпеки та попереджує вставку шкідливих сервісів в систему.

Таким чином Binder забезпечує інформування сервісів, що викликаються, в плані надання ідентифікаторів процесів, що їх викликають. Це надає змогу контролювати сервісам можливість їх використання довіреним процесам. Також Binder токени є унікальними в процесному середовищі та можуть використовуватись як маркер доступу до сервісу.

2.1.4 Застосунки та права доступу

На рівні з пісочницею знаходиться один з основних механізмів безпеки Android – права доступу застосунків на функцій системи, що зможуть в майбутньому бути використаними застосунком. До цього списку відноситься доступ до картки пам'яті, робота з камерою, використання GPS модулю та також фунціонал, що може призвести до витоку даних в мережу Інтернет або втраті грошових коштів з мобільного рахунку.

На цей рахунок в системі Android існує необхідність кожного застосунку містити в собі інформацію про функції, які він може використовувати. Ця інформація знаходиться в файлі AndroidManifest.xml всередині APK-файлу – архівного виконуваного файлу застосунків для Android, і добувається інсталятором перед встановленням застосунку з метою надати користувачеві інформацію про модулі системи, що можуть бути використаними встановлюваним застосунком. В такому випадку користувачеві необхідно погодитись із наведеним списком, після чого відбудеться інсталяція.

Такий підхід забезпечує важливу особливість – користувацькі налаштування мають більшу значимість ніж запити застосунку. Це значить, що при вимкненій користувачем функції системи застосунок ніяк не зможу отримати доступ до цієї

функції. Також, як зазначалось вище, деякі функції взагалі недоступні для застосунків (наприклад маніпуляції з SIM-картою).

2.1.5 Захист стеку

В мобільній системі Android передбачено захист від переповнення буферу та його пошкодження. Починаючи з версії 4.0 в системі присутня технологія Address space layout randomization (ASLR), що реалізує розміщення в адресному просторі образу виконуваного файлу, завантажених бібліотек, кучі та стеку випадковим чином. Завдяки такому способу розміщення елементів експлуатація багатьох типів атак значно ускладнилась, адже в такому випадку зловмиснику необхідно вгадувати адреси переходу для успішної атаки.

На рівні з цим функціонує захист від читання системного журналу ядра та захист від атак, що базуються на перезаписі секцій, що завантажені в пам'ять ELF-файлу – двійковий формат виконуваних та зв'язувальних файлів.

2.1.6 Репозиторій застосунків

Найслабшим місцем ОС Android є репозиторій застосунків Google Play. Не зважаючи на систему сповіщення про використовувані застосунком модулі системи користувач все рівно наражають на небезпеку власні пристрої.

Можливість додавання власних застосунків в репозиторій була досить легкою процедурою. Для вирішення цієї проблеми більш простим способом, ніж у iOS від Apple, де кожен застосунок піддається ручній перевірці, було створено автоматичну систему перевірки застосунків під назвою Bouncer. Цей сервіс являє собою віртуальну машину, що багаторазово запускає кожен новоопублікований

застосунок, виконує велику кількість операцій, подібних до роботи користувача з застосунком, та в кінці порівнює стан системи до використання застосунку та після.

Таким чином кількість шкідливого ПО в репозиторії Android значною мірою зменшилась. Проте існує варіант обходження Bouncer. Проаналізувавши характеристики системи-аналітика можна створити застосунок, що не буде викликати підозри у відомої зловмиснику віртуальної машини, що тестує ПЗ.

Варіант вирішення такої проблеми досить простий і полягає у створенні унікальних віртуальних середовищ для тестування кожного нового ПЗ, що унеможливить підробку.

2.2 Безпека Android застосунків

Як було сказано, основою ОС Android є ядро Linux, в яке було зроблено деякі зміни розробниками Google. Застосунки для операційної системи Android розробляються на мові Java або на досить нових мовах Scala та Kotlin. Починаючи з версії Android 1.5 був представлений набір інструментів Android NDK, який дозволяє розробляти модулі застосунків на мовах C і C++ і компілювати їх в машинний код. Застосунки поставляються у вигляді файлів спеціального формату APK, який є ZIP-архівом з певною структурою каталогів і файлів. APK-файл програми містить:

- Маніфест;
- Модулі, скомпільовані в машинний код (колективні бібліотеки .so);
- Різноманітні ресурси програми;
- DEX-файл;
- Інші необхідні файли.

Починаючи з версії Android 4.4 підтримуються два середовища виконання застосунків: Dalvik VM і ART. Слід зазначити, що процес підготовки APK-файлу не змінився, змінився тільки процес установлення застосунків в новому середовищі виконання ART. Починаючи з версії 5.0 виконавче середовище ART стало використовуватися за замовчуванням замість Dalvik VM.

Середовище виконання Java-програм Dalvik VM на Android сильно відрізняється від «звичайної» Java VM. По-перше, при компіляції Java-програми вона компілюється в байт-код віртуальної машини Dalvik, який сильно відрізняється від байт-коду віртуальної машини HotSpot. Віртуальна машина Dalvik є реєстрової, що робить її виконання на RISC-архітектурах, часто використовуваних в мобільних пристроях, більш ефективним. Також при розробці Dalvik враховувалися обмеження пам'яті в мобільних пристроях. Починаючи з версії Android 2.2 виконавча Dalvik містить JIT-компілятор, який компілює на льоту згенеровані шматки коду Java-програм в машинний код.

Стандартні бібліотеки Java були або замінені на допрацьовані бібліотеки з пакету Apache Harmony, або написані заново. При компіляції Java-програми отримується файл формату DEX (Dalvik Executable), який містить байт-код для віртуальної машини Dalvik. Формат файлу був розроблений з метою скорочення обсягу займаної пам'яті і суттєво відрізняється від стандартного формату JAR. Для виклику модулів, реалізованих в машинному коді, з Java-програм використовується інтерфейс JNI. Варто відзначити, що є можливість довантажувати машинні модулі динамічно по мережі за допомогою компонента DexClassLoader.

Батьківським процесом для всіх застосунків в ОС Android є процес Zygote. Представлений процес являє собою каркас Android-застосунку, в якому вже завантажені всі необхідні бібліотеки середовища Android, але відсутній код самогз застосунку. Запуск програми Android з точки зору операційної системи відбувається наступним чином:

1 Спочатку відбувається системний виклик fork для створення нащадка від процесу Zygote (Рисунок 2.3);
2 У новоствореному процесі відкривається файл програми яку ви запускаєте (системний виклик open).
3 Відбувається читання інформації про файлах класів (classes.dex) і ресурсів з файлу програми. Відбувається відкриття сокетів для IPC;
4 Виконується системний виклик mmap для транспортування файлів програми в пам'ять;
5 Середовище виконання робить налаштування необхідного оточення і виконує застосунок (інтерпретує байт-код Dalvik або передає управління функціям в виконуваному коді в разі ART).

На рівні ядра ОС Android кожен застосунок є окремим процесом з унікальними значеннями user / group ID, які даються йому при установці. Таким чином, кожна програма працює в своїй пісочниці. Починаючи з версії 4.3 на додаток до цієї політики безпеки додалося використання компонента мандатної контролю доступу SELinux.

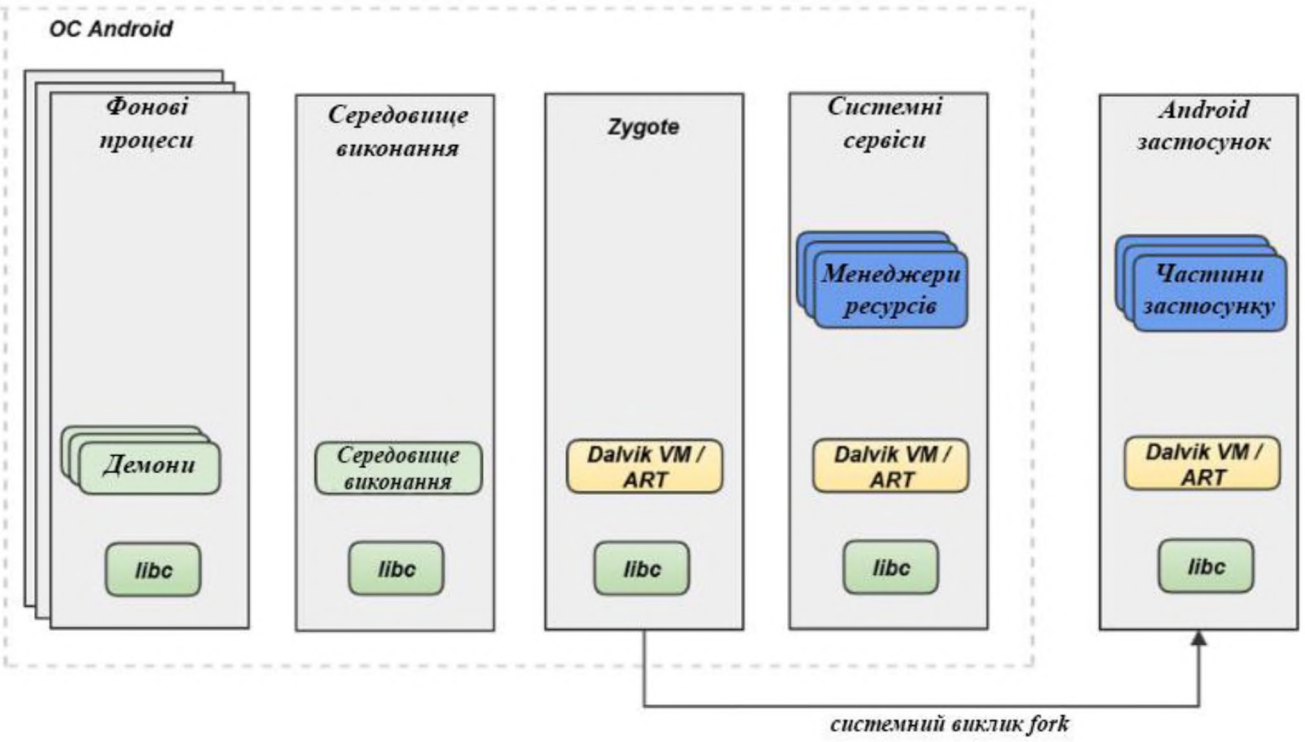

Рисунок 2.3 – Схема роботи ОС Android із застосунками

За замовчуванням застосунок обмежений в своїх можливостях і не може нічого зробити, щоб негативно вплинути на іншу програму або користувача: ні прочитати призначені для користувача дані, ні модифікувати системні програми; він також не має доступу до мережі. Для отримання доступу до різних ресурсів застосунок звертається до сервісів ОС Android. Дозволи на доступ задаються статично в файлі маніфесту застосунку і видаються з застосунком під час його роботи за мірою необхідності. Система запитує у користувача згоду на видачу цих дозволів під час встановлення або під час оновлення програми. Відповідальність за видачу доступу з додатком лежить на користувачеві, він самостійно вирішує, якому додатку давати дозволу на певні дії, а яким не давати, - під час його установки. Використання такого підходу, при якому всі дозволи видаються разом при установці застосунку, призвело до того, що користувачі стали роздавати повноваження застосункам, не замислюючись про наслідки. У свою чергу, застосунки стали запитувати все більше дозволів в міру розширення їх

функціональності. В Android 6 Marshmallow дана система замінена на іншу: застосунок запитує доступ до ресурсів у користувача під час його роботи, а користувач може або дозволити доступ, або заборонити його.

Android-застосунок складається з чотирьох видів компонентів і не містить функції main() або якоїсь іншої єдиної точки входу. Компоненти додатків взаємодіють один з одним за допомогою спеціальних повідомлень, званих Intent.

Компоненти під назвою Activity визначають призначений для користувача інтерфейс. Як правило, використовується один компонент Activity для опису одного екрану програми. Activity може запустити інше Activity, передавши параметри за допомогою механізму Intent. Під час роботи тільки одне Activity може працювати і обробляти користувальницькі дії, інші на цей час залишаються замороженими або знищуються, в залежності від обраного режиму роботи програми.

Компоненти під назвою Service займаються фоновою обробкою. Коли Activity необхідно виконати якусь операцію, наприклад завантаження файлу або програвання музики, і продовжити роботу, коли користувач перейшов в інший застосунок або згорнув поточний, застосунок запускає сервіс, мета якого – виконання цієї операції. Розробники можуть використовувати Service як застосунок-демон, який стартує під час завантаження системи. Компонент Service, як правило, підтримує RPC (Remote Procedure Call), тому інші компоненти системи можуть звертатися до нього.

Компонент Content provider забезпечує зберігання і обмін даними, використовуючи інтерфейс реляційних баз даних. Кожен Content provider містить унікальний URI для даних і обробляє запити до нього у вигляді SQL запитів (Select, Insert, Delete).

Компоненти Broadcast receiver виступають в ролі контейнерів для повідомлень від інших застосунків.

2.2.1 Вразливості ядра Linux Kernel

Головним чином вразливості, які відносяться до цієї категорії, стосуються всіх операційних систем, що побудовані на основі Linux ядра, як і ОС Android. Експлойти, які використовуються вразливості ядра, можуть отримувати дані користувача чи права адміністратора системи. Підвищивши привілеї процесу до прав адміністратора (суперкористувача) системи, шкідлива програма може відключити систему безпеки Android, вставити в існуючі програми шкідливий код і встановити руткіт. До того ж виробники різних пристроїв додають в ядро модулі для своїх продуктів; в цих модулях також можуть бути уразливості, адже на такого роду вразливості зазвичай реагують не так оперативно, як на вразливості «pure» Android в компанії Google, і вирішувати ці проблеми мають вже сторонні розробники цих модулів. Також варто відзначити, що зовсім недавно була виявлена уразливість в компоненті ashmem для взаємодії між процесами в Android.

2.2.2 Вразливості компонентів від сторонніх розробників

Останнім часом поширеною тенденцією є створення власних оболонок Android. Зазвичай такі модифіковані збірки мають ряд нововведень, додаткових застосунків, сервісів, які за замовчуванням неможливо видалити. За статистикою 60-80% вразливостей мобільних пристроїв на ОС Android можливі тільки завдяки таким стороннім оболонкам.

Наприклад в жовтні 2016 року було знайдено критичну вразливість в прошивках від Foxconn. З словами Джона Соєра, американського експерта з безпеки, debug-режим під назвою «factory test mode» таких прошивок можна було використовувати для обходу модулю SELinux, адже в цьому режимі користувачу надавалися права суперкористувача, а всі сервіси безпеки операційної системи вимикалися. Цей бекдор назвали «Pork Explosion».

2.2.3 Вразливості модулів машинних кодів

Як зазначалося вище, Android-застосунки підтримують запуск машинного коду через інтерфейс JNI. Це породжує ще одну категорію вразливостей, пов'язану з широко відомими вразливостями витоків пам'яті в низькорівневих мовах програмування (наприклад, в C і C ++). Оскільки на рівні процесів в ОС Android немає ніяких обмежень, крім тих, що накладаються ядром Linux, сторонні бібліотеки в машинних кодах можуть використовувати дозволи, видані всьому застосунку підсистемою SELinux, для здійснення шкідливої активності. Також модулі застосунку в машинних кодах використовуються авторами шкідливих програм, щоб обійти інструменти аналізу і моніторингу рівня Android. Ці модулі також можуть використовувати техніки протидії аналізу, що використовуються в звичайних програмах.

2.2.4 Вразливості механізмів міжкомпонентної взаємодії

До даної категорії відносяться вразливості, пов'язані із взаємодією між різними компонентами застосунків. Так як на рівні операційної системи застосунок обмежений пісочницею процесу, йому необхідно мати механізм доступу до компонентів ОС Android для взаємодії з ними. Це відбувається через пристрій /dev/Binder і різні інші сервіси ОС Android. Як вже говорилося вище, параметри цього доступу задаються у файлі маніфесту, у вигляді XML-файлу з дозволами. Аналіз, наведений в різноманітних статтях, вказує на недоліки цієї системи обмежень і показує шляхи їх обходу. Так, наприклад, застосунок може скористатися правами доступу іншої програми і отримати за допомогою нього дані через ICC – компілятор Intel® C++ для Android. Також існують вразливості, пов'язані зі сторонніми бібліотеками. Сторонні бібліотеки, які використовуються в застосунку, отримують той же набір обмежень, що і сам застосунок. Тому сторонні

бібліотеки можуть використовувати дозволи, видані всьому застосунку, для здійснення шкідливої активності. Програми до того ж можуть перехоплювати системні події, що пересилаються через широкомовний запит, і зберігати інформацію про вхідні дзвінки та СМС.

2.2.5 Вразливості самих застосунків

Кожна програма зберігає якісь дані про користувача. Ці дані повинні бути захищені належним чином, щоб до них не могли отримати доступ інші програми, але такий захист передбачений не завжди. Наприклад Skype в одній з версій застосунку зберігав базу даних контактів у відкритому вигляді на носії. Таким чином, контакти можна було прочитати будь-яким іншим додатком, у якого є доступ до носія інформації. Також застосунки можуть використовувати криптографічні бібліотеки з помилками або ж якісь власні реалізації алгоритмів шифрування, що зазвичай не є гарною ідеєю. До того ж не всі програми мають якісно реалізовану аутентифікацію і авторизацію користувача. Крім цього, застосунки можуть дозволяти SQL-ін'єкції і схильні до атак XSS. Також варто відзначити, що більшість розроблюваних застосунків написані на Java без використання будь-якого захисту для бінарного коду, а байт-код Java, як відомо, легко піддається дизасемблюванню і аналізу. Варто зазначити, що до цієї категорії вразливостей відноситься також відомий список Mobile OWASP-10.

2.2.6 Вразливості вбудованих сервісів та бібліотек

Стандартний набір бібліотек і сервісів, що працюють в Android, також містить вразливості. Наприклад, нещодавно була виявлена вразливість Stagefright в бібліотеці для відображення відео в MMS-повідомленнях, яку використовували

всі версії Android, починаючи з 2.2. Пізніше була виявлена вразливість в компоненті MediaServer, якій піддаються всі версії Android с 2.3 до 5.1. У статті Asaf Shabtai, Dudu Mimran, Yuval Elovici під назвою «Evaluation of Security Solutions for Android Systems» показані вразливості рантайм-середовища Dalvik: запустивши велику кількість процесів в системі, можна домогтися того, що подальший процес запуститься з правами адміністратора через проблем обмеженості оперативної пам'яті та помилок OutOfMemory.

2.2.7 Проблеми Інтернет-джерел

Поширення Android-застосунків відбувається через багато інших джерел крім офіційного магазину застосунків. Оскільки Android-застосунки написані в основному на Java, то вони легко піддаються зворотній розробці та перепакування з використанням шкідливого коду. Крім того, пісочницю аналізу застосунків Bouncer, яка використовується в офіційному каталозі, легко обійти. Тому і в самому офіційному магазині міститься велика кількість шкідливих програм. Крім цього, Android підтримує віддалене встановлення застосунків через Google Play на пристрої, пов'язані з Google-аккаунтом. Таким чином, якщо зламати обліковий запис Google для пристрою, можна встановити з Google Play шкідливий застосунок, який було попередньо завантажено зловмисником в магазин застосунків. При цьому на екрані мобільного телефону не потрібно отримувати будь-яких підтверджень цих дій, оскільки вони запитуються у вікні браузера і застосунок встановлюється на телефон в фоновому режимі при отриманні доступу до Інтернету. Також до цієї категорії вразливостей відноситься використання соціальної інженерії, коли для продовження роботи пропонують встановити застосунок з неавторизованого джерела.

2.3 Безпека iOS

На сьогоднішній день мобільна операційна система від компанії Apple під назвою iOS є самою безпечною на фоні інших мобільних систем. З року в рік нові розробки забезпечують все більшу безпеку приватних даних на мобільному пристрої, а можливі бекдори в мережі стають сенсаціями, аніж буденними справами, чого не кажеш про Android.

На Рисунку 2.4 зображено схему роботи моделі безпеки операційної системи в парі з hardware частиною.

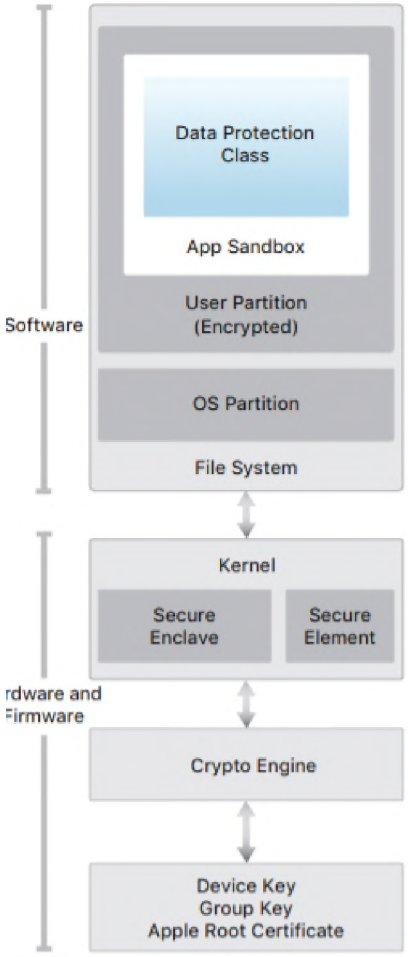

Рисунок 2.4 – Діаграма архітектури безпеки операційної системи iOS

Головним чином вся безпека цієї ОС розбита на наступні модулі:

- Система безпеки: інтегроване та захищене програмне та апаратне забезпечення;
- Шифрування та захист даних: архітектура та дизайн, що захищають дані користувача, якщо пристрій загублено або викрадено, або якщо неавторизований користувач намагається використати або змінити приватні дані;
- Захист застосунків: системи, які дозволяють працювати захищено, при цьому не компрометуючи цілісність платформи;
- Мережева безпека: стандартизовані мережеві протоколи, що забезпечують безпечну автентифікацію та шифрування даних при їх передачі;
- Apple Pay: реалізація безпечних платежів від компанії Apple;
- Інтернет-послуги: мережева інфраструктура Apple для обміну повідомленнями, синхронізація та резервне копіювання;
- Управління паролем користувача: обмеження паролем та доступ до паролю з інших авторизованих джерел;
- Керування пристроями: методи, які дозволяють керувати пристроями iOS, запобігати неавторизованому використанню та вмикати віддалену очистку пам'яті, якщо пристрій втрачено або вкрадено;
- Контроль конфіденційності: можливості iOS, які можна використовувати для контролю доступу на Служб локації та персональних даних користувача.

2.3.1 Безпека системи iOS

Тісна інтеграція обладнання, програмного забезпечення та сервісів на пристроях iOS гарантує, що кожен компонент системи є довіреним і перевіряється системою в цілому. Від початкового завантаження та оновлення програмного забезпечення iOS до роботи сторонніх програми, кожен крок аналізується і перевіряється, щоб забезпечити оптимальну роботу апаратного і програмного забезпечення і використовують ресурси правильно.

Відразу після включення пристрою в роботу вступають сервіси забезпечення захищеності пристрою. В першу чергу процесор програм виконує код з пам'яті, що доступна тільки для читання, під назвою Boot ROM. Цей незмінний код, що відомий як апаратний корінь довіри, закладається ще в процесі виготовлення мікросхеми і вважається безумовно достовірним та надійним. Код цього Boot ROM несе в собі відкритий ключ рутового сертифікату Apple, що використовується для підтвердження того, що завантажувач системи iBoot є сертифікованим Apple. Коли iBoot виконає всі свої задачі, він перевіряє саме ядро та запускає його. В останніх моделях пристрою Apple відбувається на одну перевірку менше, а саме на пристроях з процесорами моделей S1, A9 та старшими моделями додатково виконується перевірка та завантаження низькорівневого (Low-Level Bootloader – LLB) завантажувача, який в свою чергу завантажує та перевіряє iBoot. Таким чином організовується так званий ланцюг довіри, в якому кожна ланка переконується в тому, що наступна ланка сертифікована, цілісна та підписана компанією Apple.

В тому випадку, якщо LLB чи iBoot не вдалося завантажитися, тоді пристрій переходить в режим оновлення – DFU, що також називається режим відновлення.

В той же час Boot Progress Register обмежує доступ до персональних даних користувача при переході до режиму відновлення.

На пристроях із стільниковим доступом також використовується підсистема базової лінії зв'язку, що має власний аналогічний процес захищеного завантаження

з підписаними програмним забезпеченням та ключами, які перевіряються основним процесором.

Співпроцесор Secure Enclave також використовує захищений процес завантаження, який гарантує, що його окреме програмне забезпечення перевірено та підписано компанією Apple.

Secure Enclave - це співпроцесор, вбудований в процесори Apple T1, Apple S2, Apple S3, Apple A7 і більш нові версії процесорів серії A. Secure Enclave використовує шифровану пам'ять і включає в себе апаратний генератор випадкових чисел. Secure Enclave забезпечує виконання всіх криптографічних операцій для управління ключами захисту даних і гарантує цілісність системи захисту даних навіть у разі порушення безпеки ядра. Обмін даними між Secure Enclave і процесором програм обмежений поштовим ящиком, який керується за допомогою переривань, і загальними буферами даних в пам'яті.

Під час завантаження пристрою створюється динамічний ключ, який пов'язаний з UID пристрою. За допомогою цього ключа виконується шифрування області пам'яті пристрою, яка призначена для Secure Enclave. На всіх процесорах, окрім Apple A7, динамічний ключ також використовується для аутентифікації пам'яті Secure Enclave. У процесорі Apple A11 використовується дерево цілісності, що запобігає повторній передачі критичної для безпеки пам'яті Secure Enclave. Аутентифікація цього дерева виконується за допомогою динамічного ключа і значень nonce, що зберігаються в пам'яті SRAM на мікросхемі.

Дані, збережені в файловій системі Secure Enclave, зашифруються за допомогою ключа, пов’язаного з ідентифікатором UID та лічильником антиповторення (anti-replay). Лічильник антиповторення зберігається записаний на спеціальній інтегральній схемі (IC), що не володіє енергонезалежною пам'яттю.

На пристроях з A12 та S4 SoC Secure Enclave суміщено з захищеним сховищем інтегральної схеми (IC), що використовується для зберігання лічильника антиповторів. Безпечне сховище IC розроблено з незмінним кодом ROM, апаратним генератором випадкових чисел, криптографічними сервісами та фізичним виявленням вторгнення. Щоб читати та оновлювати лічильники, Secure

Enclave та IC сховище використовують захищений протокол, який забезпечує винятковий (недоступний для інших сервісів) доступ до лічильників.

Служби Anti-replay у Secure Enclave використовуються для відкликання даних про події, що позначають межі антиреплікації, включаючи, але не обмежуючись такими подіями:

- Зміна паролю;
- Touch ID або Face ID ввімкнення або вимкнення;
- Додавання та видалення зразку відбитків пальців;
- Скидання Face ID;
- Додавання та видалення карт Apple Pay;
- Видалення всього контенту та налаштувань з пристрою.

Після того, як ядро iOS завершить ініціалізацію, буде активовано Kernel Integrity Protection (KIP), щоб запобігти модифікації коду ядра та драйверів. Контролер пам'яті забезпечує завантаження в область захищеної фізичної пам'яті, яку використовує iBoot, ядра та розширень ядра. Після завершення завантаження контролер пам'яті забороняє запис до захищеної області фізичної пам'яті. Крім того модуль керування пам'яттю процесора застсосунків (MMU) налаштовано таким чином, щоб запобігти маппінгу привілейованого коду з фізичної пам'яті за межі області захищеної пам'яті та запобігти запису відображення фізичної пам'яті в межах області пам'яті ядра.

Touch ID – це система зняття та первірки відбитків пальців, що робить захищений доступ до iPhone та iPad швидким та простішим. Ця технологія зчитує дані відбитків пальців з будь-якого кута і вивчає додаткові відомості про відбитки пальців користувача з плином часу, при цьому датчик продовжує розширювати карту відбитків пальців, оскільки додаткові дублюючі вузли ідентифікуються з кожним використанням.

Face ID – аналогічна Touch ID система, яка в свою чергу базується на трьох вимірному знімку обличчя та надійно розблоковує пристрої Apple, які мають цю функцію. Він забезпечує інтуїтивно зрозумілу автентифікацію, що забезпечується за допомогою систем камер TrueDepth, які використовують передові технології для точного визначення геометрії вашого обличчя. Ідентифікатор особи використовує нейронні мережі для визначення погляду, спрямованого на телефон, співпадінням та запобігання підміни (наприклад фотографією), тому ви можете швидко розблокувати телефон. Ідентифікатор особи автоматично адаптується до змін у вашому зовнішньому вигляді та ретельно захищає конфіденційність та безпеку ваших біометричних даних.

З вимкненими Face ID та Touch ID, коли пристрій блокується, ключі для найвищого класу захисту даних, які зберігаються в Secure Enclave, відключаються. Файли та елементи Keychain (ланцюг зашифрованих даних) в цьому класі недоступні, поки ви не розблокуєте пристрій, ввівши пароль.

З ввімкненими Touch ID або Face ID, ключі не відключаютсья, коли пристрій блокується; замість цього вони зашифровані ключем, який надається підсистемі Touch ID або Face ID всередині Secure Enclave. Коли ви намагаєтесь розблокувати пристрій, якщо пристрій виявить успішний збіг, він забезпечує ключ для розпакування ключів захисту даних, а пристрій розблоковується. Цей процес забезпечує додатковий захист, вимагаючи співпраці між підрозділами захисту даних та Touch ID або ідентифікаторами Face ID, щоб розблокувати пристрій.

Коли пристрій перезавантажиться, ключі, необхідні для Touch ID або Face ID, щоб розблокувати пристрій, втрачаються; вони вважаються відхиленими сервісом Secure Enclave після виконання будь-яких умов, які потребують введення пароля (наприклад, після відсутності спроб розблокування протягом 48 годин або після п'яти невдалих спроб розблокування з використанням Touch ID або Face ID).

Сторонні застосунки можуть використовувати API, надані системою, для проходження автентифікації за допомогою ідентифікатора Touch ID або ідентифікатора Face ID чи коду доступу, а програми, які підтримують Touch ID, автоматично підтримують Face ID без будь-яких змін. При використанні

ідентифікатора Touch ID або ідентифікатора Face ID, застосунку повідомляється лише про те, чи автентифікація була успішною; він не може отримати доступ до Touch ID та Face ID даних чи даних, пов'язаних із зареєстрованим користувачем. Елементи KeyChain також можуть бути захищені за допомогою ідентифікатора Touch ID або Face ID, доступ до яких буде видано сервісом Secure Enclave лише за допомогою успішного проходження відповідності Touch ID чи Face ID або коду доступу пристрою. Розробники застоунків мають API, щоб підтвердити, що користувач встановив пароль, перш ніж вимагати ідентифікатор Touch ID або Face ID, або пароль для розблокування елементів Keychain. Розробники програм можуть робити наступне:

- Дозволяти API автентифікації не поверталися до запиту пароля програми або коду доступу до пристрою. Вони можуть запитувати, чи є користувач зареєстрованим, дозволяючи використовувати Touch ID або ідентифікатор особи як другий фактор у застосунках, що мають бути захищеними.
- Створювати та використовувати ключі ECC у Secure Enclave, які можна захистити за допомогою Touch ID або Face ID. Операції з цими ключами завжди виконуються всередині Secure Enclave тільки після того, як підтверджується авторизований доступ до їх використання.

Також для розблокування можна використовувати самий примітивний пароль доступу або іншими словами – пасскод.

Пасскод захищає ваш телефон від сторонніх і активує захист файлів. Без встановлення пасскоду ви можете віддалено видалити всі дані з пристрою. З встановленим пасскодом файли зашифровані і без паролю прочитати їх неможливо.

Пасскод шифрується ключем пристрою, перебір можливий тільки на iPhone, де його встановили.

Для захисту від перебору пасскод зашифрований кілька разів, одна перевірка правильності займає приблизно 80 мілісекунд. Щоб апаратно перебрати всі

комбінації 6 символьного пасскоду непотрібно п'ять з половиною років, а стандартного 4-х числового 14 хвилин.

Крім цього, iOS збільшує час перевірки після кожного помилкового введення. Користувачі з надзвичайно секретними даними включають опцію повного стирання телефону після 10 невдалих спроб.

2.3.2 Шифрування та захист даних

В iOS вбудований механізм шифрування файлів на диску пристрою. Механізм дозволяє віддалено і швидко очистити файлову систему, захищає ваші дані при встановленні флеш-пам'яті з вашого пристрою в інший. Якщо встановлений пасскод, прочитати вміст файлів без нього неможливо.

Захист файлів реалізований через ієрархію криптографічних ключів.

iOS присвоює кожному файлу 256 бітний AES ключ. Дані записуються на диск тільки в зашифрованому вигляді.

AES - симетричний алгоритм шифрування. Уряд США використовує AES з ключем 256 біт для зберігання секретних документів. Симетричність означає, що для шифрування і розшифровки даних використовується один і той же ключ.

В iOS існує кілька класів захисту файлів. Кожен клас захисту має свій крипто-ключ.

Файловий ключ шифрується ключом класу. Зашифрований файловий ключ зберігається в метаданих файлу. Метадані всіх файлів шифруються ключем файлової системи.

Ключ файлової системи створюється випадковим чином при першій установці iOS або очищенні пристрою. Ключ зберігається в Очищуваному сховищі. Коли користувач очищає пристрій опцією «Erase all content and settings» або віддалено через iCloud, ключ файлової системи видаляється. Тепер всі файли

недоступні, система не може розшифрувати метадані, в яких зберігається ключ файлу.

Ключ класу, в свою чергу, зашифрований унікальним ключем пристрою і пасскодом. Таку схему шифрування зображено на Рисунку 2.5.

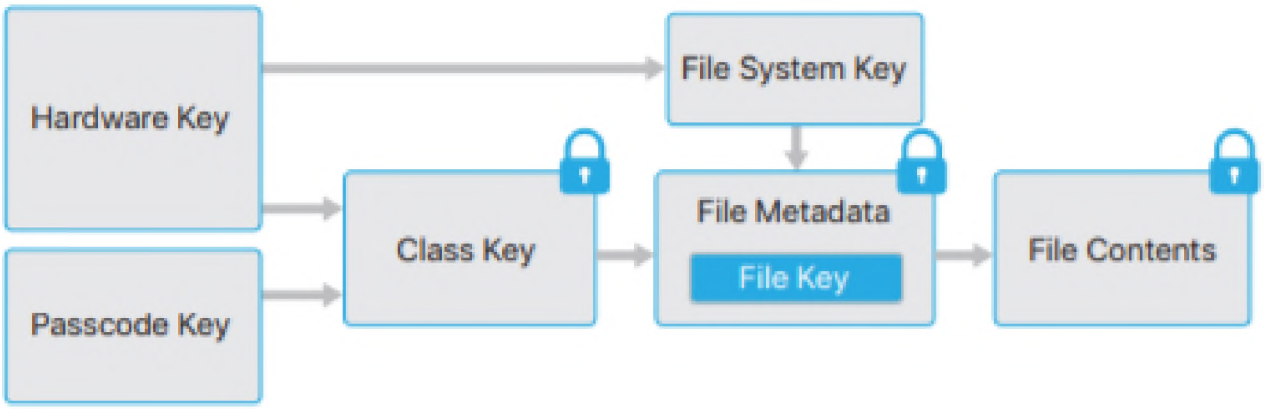

Рисунок 2.5 – Схема шифрування даних в iOS

Така схема гнучка і продуктивна. Наприклад, щоб змінити клас захисту файлу, досить повторити шифрування ключа файлу. А при зміні пасскоду достатньо повторного шифрування ключа класу захисту.

Класи захисту даних

При створенні файлу сторонні застосунки вибирають йому Клас захисту.

Кращим варіантом є використання повного захисту (NSFileProtectionComplete).

Ключ класу «Повний захист» зашифрований пасскодом і ключем пристрою. Через 10 секунд після блокування пристрою, розшифрований ключ класу стирається з пам'яті. Всі файли цього класу стають недоступні, поки користувач не введе пасскод знову.

Стандартний поштовий застосунок для листів і вкладень використовує Повний захист. Всі дані про місцезнаходження користувача зберігаються в файлах з повним захистом.

Іншим, менш захищеним варіантом є використання класу «Не захищений, якщо відкритий» (NSFileProtectionCompleteUnlessOpen).

Іноді потрібно записувати файли, коли пристрій заблоковано, наприклад, при скачуванні у фоновому режимі. Для цього існує Клас не захищений, якщо відкритий.

Для файлів, захищених цим класом, крім Ключа файлу, створюється пара з закритого і публічного ключа. З закритого ключа файлу і публічного ключа класу генерується загальний ключ (shared secret). Публічний ключ класу доступний без пасскода. Ключ файлу шифрується загальним ключем і зберігається в метаданих разом з публічним ключем файлу. Закритий ключ файлу видаляється. Як тільки запис в файл закінчена, ключ файлу стирається з пам'яті. Щоб відкрити файл заново, тимчасовий ключ відтворюється з закритого ключа класу і публічного ключа файлу. Закритий ключ класу вже захищений пасскодом.

Іншими словами, файли класу «не захищений, якщо відкритий» можуть бути створені в тлі, але після завершення запису для доступу до них потрібно пасскод.

Наступним варіантом є клас «захищений до першого входу» (NSFileProtectionCompleteUntilFirstUserAuthentication).

Працює аналогічно повному захисту, тільки розшифрований ключ файлу не видаляється з пам'яті при блокуванні пристрою. Цей клас захищає дані від атак, пов’язаних із перезавантаженням пристрою. Клас використовується за умовчанням в сторонніх застосунках.

І останнім класом є тип «без захисту» (NSFileProtectionNone).

Ключ класу без захисту зашифрований тільки ключем пристрою. Файли, захищені цим класом, не можна прочитати на іншому пристрої і можна швидко зробити недоступними віддалено.

На апаратному рівні в цій операційній системі також передбачена підтримка шифрування.

Постійні шифрування і розшифровування всіх файлів сильно гальмувало б роботу системи і зменшило час роботи від батареї. Тому в кожному пристрої Apple на iOS шифрування реалізовано на апаратному рівні.

На шляху між оперативною пам'яттю і флеш-картою вбудований чіп AES-256 шифрування. Процесор передає йому ключ файлу, а після цього дані прозоро і швидко шифруються і дешифруються при доступі до файлу.

Унікальний ключ пристрою (UID) і ключ групи пристроїв (GID) це 256 бітні AES ключі, вшиті в процесор при виробництві. Програмне забезпечення не має доступу до цих ключів, їм доступні лише результати шифрування та дешифрування цими ключами. UID унікальний для кожного пристрою і не зберігається у Apple або її постачальників. GID збігається для всіх пристроїв з одним типом процесору (наприклад, для всіх пристроїв з A7 чіпом). GID використовується як додатковий захист при установці і відновленні системи. Прошивання цих ключів на чіпі дозволяє уникнути їх підміни, обходу або отримання доступу повз апаратну реалізацію AES-шифрування.

UID дозволяє криптографічно прив'язати дані до конкретного пристрою. Наприклад, ієрархія ключів шифрування файлової системи включає в себе UID. Якщо переставити чіп пам'яті з одного пристрою на інший, файли будуть недоступні.

Всі інші крипто-ключі належать системним генераторам випадкових чисел. Ентропія для ключів рахується з часу виконання системних запитів при завантаженні і даних сенсорів пристрою після цього виконання.

Безпечне видалення ключів настільки ж важливе, як і їх створення. Це особливо складно виконати на флеш-накопичувачах, де через зношування дані можуть дублюватися в декількох місцях. Тому iOS пристрої містять Очищуване сховище (Effaceable Storage). Ця функція дозволяє стирати невеликі блоки даних з пристрою зберігання на найнижчому рівні системи.

2.3.3 Безпека застосунків

В iOS реалізована багаторівнева система захисту для гарантії того, що програми підписані і перевірені, а також запускаються в так званій «пісочниці» для захисту персональних даних користувача. Ці елементи створюють стабільну, безпечну платформу для роботи програм, дозволяючи тисячам розробників пропонувати сотні тисяч програм для iOS без шкоди для цілісності системи. А користувачі можуть використовувати ці програми на своїх пристроях iOS, не побоюючись вірусів, шкідливих програм або інших атак.

Після запуску ядра iOS воно контролює, який користувач може обробляти процеси та додатки. Щоб переконатися, що всі програми надходять із відомих та затверджених джерел, і вони не були підроблені, iOS вимагає підписати всі виконувані коди за допомогою сертифіката, виданого Apple. Програми, що постачаються разом із пристроєм, такі як Mail і Safari, підписані компанією Apple. Сторонні програми також повинні бути перевірені та підписані, використовуючи сертифікат, виданий Apple. Обов'язковий підпис коду розширює поняття ланцюжка довіри з ОС до застосунків та забороняє стороннім програмам завантажувати непідписані ресурси коду або використовувати самостійно змінюючийся код.

Щоб розробляти та встановлювати застосунки на пристроях iOS, розробники повинні зареєструватися в Apple і приєднатися до програми розробника Apple. Ідентифікація кожного розробника, незалежно від того, чи це особа, чи компанія, перевіряється Apple перед їх видачею сертифікату. Цей сертифікат дозволяє розробникам підписувати застосунки та передавати їх в App Store для розповсюдження. Як наслідок, всі застосунки у магазині App Store є представлені визначеною особою чи організацією, що служить стримуючим фактором до створення шкідливих програм. Вони також були переглянуті компанією Apple, щоб гарантувати, що вони працюють, як описано, і не містять явних помилок або інших проблем. Окрім технології, яку вже обговорювали, цей процес закупівель дає споживачам впевненість у якості додатків, які вони купують.

iOS дозволяє розробникам вписувати фреймворки в свої застосунки, які можуть використовуватися самим застосунком або розширеннями, вбудованими в застосунок. Щоб захистити систему та інші застосунки від завантаження стороннього коду усередину їх адресного простору, система виконує перевірку підпису коду всіх динамічних бібліотек, які процес виконує на час запуску. Ця перевірка виконується за допомогою ідентифікатора команди (ідентифікатор групи), який витягується з сертифіката, виданого компанією Apple.

Ідентифікатор команди – це 10-значний буквено-цифровий рядок; наприклад, 1A2B3C4D5F. Програма може зв'язуватися з будь-якою бібліотекою платформи, що поставляється разом із системою або будь-якою бібліотекою з тим самим ідентифікатором команди в його підписі коду як головному виконуваному файлі. Оскільки завантаження виконуваних файлів у складі системи не має ідентифікатора команди, вони можуть зв'язуватися лише з бібліотеками, які постачаються разом із системою.

Підприємства також мають можливість створювати внутрішні програми для використання в рамках своєї організації та поширювати їх своїм співробітникам. Підприємства та організації можуть подати заявку на програму Apple Enterprise Developer Enterprise (ADEP) з номером D-U-N-S. Apple затверджує заявників після перевірки їх особи та відповідності. Після того, як організація стає членом ADEP, вона може зареєструватися, щоб отримати профіль Provisioning, який дозволяє внутрішнім застосункам працювати на пристроях, які він надає. Користувачам необхідно встановити профіль надання для запуску внутрішніх додатків. Це гарантує, що лише цільові користувачі організації зможуть завантажувати застосунки на свої пристрої iOS. Програми, встановлені через MDM, неявно надійні, оскільки зв'язок між організацією та пристроєм вже встановлений. В іншому випадку користувачі мають схвалити профіль для надання застосунків у налаштуваннях. Організації можуть заборонити користувачам схвалювати застосунки від невідомих розробників. При першому запуску будь-якого корпоративного застосунку пристрій має отримати позитивне підтвердження від Apple про те, що програма може працювати.

На відміну від інших мобільних платформ, iOS не дозволяє користувачам встановлювати потенційно шкідливі непідписані програми з веб-сайтів або запускати ненадійний код. В runtime виконується перевірка кодів підпису всіх сторінок виконуваної пам'яті на те, чи однакові вони з тим, що завантажувалося в пам’ять, щоб переконатися, що застосунок не було змінено з моменту його встановлення або останнього оновлення.

Іншим вадливим аспектом є процес забезпечення захищеності під час виконання застосунків.

Після того, як застосунок перевіряється на походження із затвердженого джерела, iOS застосовує заходи безпеки, розроблені таким чином, щоб запобігти небезпеці використання інших програм або решти системи встановленим застосунком.

Усі сторонні програми є "ізольованими" (принцип пісочниці), тому вони не мають доступу до файлів, що зберігаються іншими програмами, або до внесення змін у пристрій. Це забороняє програмам збирати або змінювати інформацію, що зберігається іншими програмами. Кожен застосунок має унікальний домашній каталог для своїх файлів, який випадковим чином призначається, коли виконується встановлення застосунку. Якщо третій сторонній програмі потрібно отримати доступ до інформації, що не знаходиться у власному каталозі, це робиться лише за допомогою служб, явно наданих системою iOS.

Системні файли та ресурси також захищені від застосунків користувача. Більшість частина операційної системи iOS працює як непривілейований користувач під назвою "mobile", як і всі сторонні програми. Весь розділ ОС монтується лише для читання. Непотрібні інструменти, такі як служби віддаленого входу, не включені в системне програмне забезпечення, а API не дозволяють застосункам перевищувати власні права на зміну інших програм або самої iOS.

Доступ сторонніх застосунків до інформації про користувача та функцій, таких як iCloud та розширень (extensions), регулюється за допомогою оголошених прав. Дозволи – це пари ключ-значення, які вносяться в застосунок, і підлягають перевірці автентичності за факторами виконання часу, такими як ідентифікатор

користувача UNIX. Оскільки права підписані цифровою формою, їх неможливо змінити. Правозастосування широко використовуються системними застосунками та демонами (віртуальними образами) для виконання певних привілейованих операцій, які в іншому випадку вимагатимуть, щоб процес запускався як root (суперкористувач). Це значно знижує потенціал ескалації привілеїв через компромісний системний застосунок або образ.

Крім того, застосунки можуть виконувати фонову обробку даних лише за допомогою API, наданих системою. Це дає змогу застосункам продовжувати функціонувати без зниження продуктивності або значного впливу на акумулятор пристрою.

Рандомізація адресного простору (ASLR) захищає від експлуатації помилок кодування пам'яті. Вбудовані застосунки використовують ASLR, щоб забезпечити випадкове розташування власного коду по всій області після запуску. Випадкове розташування адрес пам'яті виконуваного коду, системних бібліотек та пов'язаних програмних конструкцій зменшує вірогідність багатьох складних експлойтів. Наприклад, спроба повернення до libc намагається змусити пристрій виконувати шкідливий код, маніпулюючи адресами пам'яті стеків та системних бібліотек. Рандомізація розміщення цих елементів робить атаку набагато складнішою, особливо на кількох пристроях. Xcode, середовище розробки iOS, автоматично компілює сторонні програми з підтримкою ASLR.

Подальший захист забезпечується iOS за допомогою функції ARM’s Execute Never (XN), яка позначає сторінки пам'яті як невиконувані. Сторінки пам'яті, що позначені з правами дозволу записування і виконування, можуть бути використані лише застосунками під жорсткими умовами контролю: ядро перевіряє наявність права динамічного підписування коду на Apple. Навіть після цього можна зробити лише один виклик mmap для запиту на виконування та запис цієї сторінки, яка дається за рандомізованою адресою. Safari використовує цю функцію для свого JavaScript JIT-компілятора.

Іншою корисною функцією ОС є можливість використання розширень. iOS дозволяє застосункам надавати функціональні можливості для інших застосунків,

надаючи розширення. Розширення – спеціально підписані виконувані файли, упаковані в застосунок. Система автоматично визначає розширення в установлений час і робить їх доступними для інших застосунків за допомогою відповідної системи.

Системна область, що підтримує розширення, називається точкою розширення. Кожна розширювальна точка забезпечує API та впроваджує політику для цієї області. Система визначає, які розширення доступні на основі правил узгодження з точкою розширення. Система автоматично запускає процеси розширення, якщо це необхідно, і керує їх життям. Права можуть бути використані для обмеження доступності розширення до певних системних зстосунків. Наприклад, віджет перегляду «Today» відображається лише в Центрі сповіщень, а розширення спільного доступу доступне лише з області «Спільний доступ». Точки розширення – віджети Today, Share, Custom actions, редагування фотографій, Document Provider та користувацькі клавіатури.

Розширення працюють у власному адресному просторі. Зв'язок між розширенням та застосунком, з якого воно було активоване, забезпечується використанням міжпроцесних комунікацій, опосередкованих системою. Вони не мають доступу до файлів чи простору в пам'яті. Розширення розроблені таким чином, щоб бути ізольованими один від одного, від їх вмісту та від застосунків, які їх використовують. Вони є ізольованими, як будь-який інший сторонній застосунок, і їх контейнер даних розташований окремо від контейнера, що містить застосунок. Проте вони мають такий самий доступ до елементів керування конфіденційністю, що і застосунок контейнера. Тому, якщо користувач надає доступ до контактів для застосунку, цей грант буде поширюватися на розширення, вбудовані в застосунки, а не на розширення, активовані застосунком.

Для пристроїв, зареєстрованих у рішенні MDM, розширення документів та клавіатури повинні відповідати правилам керованого відкритого доступу. Наприклад, рішення MDM може перешкодити користувачеві експортувати документи з керованої програми в некерований постачальник документів або використовувати некеровану клавіатуру з керованою програмою. Окрім того,

розробники застосунків можуть заборонити використання сторонніх розширень клавіатури у своїх програмах.

Також в цій ОС необхідну увагу приділено захисту даних мобільних застосунків. Комплект програмного забезпечення для розробників iOS (SDK) пропонує повний набір API-інтерфейсів, які допомагають стороннім і внутрішнім розробникам адаптувати захист даних на власний манер та допомагають забезпечити найвищий рівень захисту в своїх застосунках. Захист даних доступний для API-файлів і баз даних, включаючи NSFileManager, CoreData, NSData та SQLite.

Базу даних застосунку електронної пошти (включаючи вкладення), керовані книги, закладки Safari, образи запуску застосунків та дані про місцезнаходження також зберігаються за допомогою шифрування, а ключі захищаються кодом користувача на своєму пристрої. Календар (за винятком вкладень), контакти, нагадування, примітки, повідомлення та фотографії також підтримують реалізацію захисту даних за типом «захищений до першої перевірки автентичності користувача».

Також належну увагу тут приділено захисту від аксесуарів, що працюють з пристроями.

Програма ліцензування "Made for iPhone", "iPad" та "iPod touch" (MFi) надає додатковим виробникам застосунків доступ до протоколу iPod Access Components (iAP) та необхідних допоміжних апаратних компонентів.

Коли аксесуар MFi комунікує з пристроєм iOS за допомогою роз'єму Lightning або через Bluetooth, пристрій запитує у аксесуару підтвердження того, що він є сертифікованим Apple до роботи з Apple технікою, відсилаючи запит на отримання сертифікату Apple, який підтверджено пристроєм. Після цього пристрій надсилає виклик, на який аксесуар повинен відповісти підписаною відповіддю.

Цей процес повністю обробляється користувальницькою інтегральною схемою (IC), яку Apple постачає затвердженим виробникам аксесуарів та є прозорою для самого аксесуара.

Висновки до розділу 2

В даному розділі було проаналізовано захищеність мобільних платформ та механізми захисту, які забезпечуються для застосунків при розробці на цих платформах. Таким чином після проведеного аналізу iOS платформа є більш захищеною за Android систему з дуже щільною міжкомпонентною взаємодією елементів системи.

3 АРХІТЕКТУРА СИСТЕМИ ЗАХИСТУ МОБІЛЬНОГО ЗАСТОСУНКУ

При розробці мобільного застосунку варто враховувати всі можливі варіанти привабливості для третіх персон інформації, що циркулює в програмі. Ступінь важливості та чутливості цих даних залежить від сфери використання розроблюваного застосунку, а все рівно навіть звичайний пароль аутентифікації в застосунок потребує опрацювання алгоритму його захисту.

В першу чергу головним вибором захисту мобільного застосунку є винесення частини функціоналу за межі локального сховища, адже мінімізація некерованого доступу до інформації на пристрої покладе початок максимальній захищеності застосунку.

З одного боку використання виключно локального варіанту застосунку набагато спрощує розробку, але якщо мова йде про безпеку, то в цьому випадку набагато легше контролювати дані, які знаходяться на власних серверах та доступ до яких корегується напряму розробником, а не через оновлення та патчі застосунків, які користувач може пропустити чи відмовити в їх встановленні.

Серед основних видів атаки на мобільні застосунки розрізняють наступні:

- Декомпіляція файлу застосунку (.apk файли у випадку Anroid системи чи .ipa файли при роботі застосунку на базі iOS від Apple) та наступний розбір локально збережених даних застосунку. Захист застосунку такого роду в більшій мірі залежить від розробника застосунку, а не від провайдера платформи, не зважаючи на всі спроби великих платформ максимізувати безпеку застосунків, що розробляються для їх операційних систем;
- Перехоплення даних по незахищених каналах мережі (MITM-атаки). Як було зазначено вище, рекомендується використовувати архітектуру клієнт-серверного застосунку. За статистикою більшість застосунків мають саме такий принцип роботи і відповідно до цього такі застосунки передають по

мережі великі масиви даних. В критичному випадку така передача здійснюється через відкриті канали, але також незважаючи на пропаганду використання HTTPS-протоколу передачі даних, не варто покладатися на єдиний рубіж – захищений канал передачі відкритих до читання даних;

- Рутування пристроїв та наступна атака на застосунок і застосовувані в ньому алгоритми через зовнішні debug-інструменти.

3.1 Організація локальної захищеності мобільного застосунку

Не зважаючи на обрану модель застосунку необхідно забезпечити захищеність даних, що локально розміщені на пристрої та використовуються застосунком.

Головним чином захист застосунку на пристрої зводиться до забезпечення захищеності критично важливих даних користувача (абревіатура КВД). До цієї групи даних відносяться всякого роду відомості про користувача, його дата народження, адреса проживання, його приватні дані, такі як паролі, інформація про банківські карти, номери замовлень.

Першою і самою головною проблемою захисту є використання незахищених локальних сховищ. Часто зустрічається збереження даних в слабозахищених сховищах, далі буде приведено варіанти реалізації захищеного сховища даних.

Також можна зустріти такий випадок створення розробником загрози, як зберігання КВД напряму у коді. Прикладами такого є зберігання паролів підключення до серверів чи солей до паролів у вигляді статичних змінних, що при декомпіляції можуть бути легко розшифровані.

Таким чином зберігати таку інформацію в коді не можна.

Аналогічним випадком є використання алгоритмів зі зберіганням приватного ключа в коді програми. Для рішення такої проблеми варто використовувати сучасні симетричні алгоритми з одноразовими ключами, що випадково генеруються та є

стійкими до злому методом перебору, або виносити приватний ключ за межі застосунку та використовувати вже асиметричні приватні ключі.

Наступною проблемою використання приватних ключів є збереження їх на сервері. Таким чином сервер може розшифровувати дані користувача, а це в свою чергу дозволяє при зломі серверу отримувати доступ до приватних даних користувача. З іншого боку проста доступність даних користувача на сервері порушує приватність даних.

Рішенням цієї проблеми вважається не використання без лишньої необхідності чи дозволу користувача (у вигляді ліцензійного погодження) ключів, що будуть доступні на сервері застосунку, а паролі та ключі на сервер необхідно відправляти вже у вигляді хешу.

Іншим хибним кроком реалізації безпеки застосунку є використання самописних алгоритмів шифрування. Це несе за собою можливість легкого дешифрування даних, не залежно від «високої компетентності» розробників «нового» алгоритму. На практиці важливо використовувати перевірені часом та спробами злому алгоритми.

В деяких випадках розробники покладаючись на захищеність сховища даних зберігають там критичні дані користувача у відкритому вигляді, що робити не рекомендується без використання додаткового шифрування, адже у випадку рутування пристрою захищеність таких сховищ залишається під питанням.

Також не рекомендується використання веб-движків (вбудований браузер) для маніпуляцій з критичними даними, так як рівень захищеності в таких випадках знижується через незнання механізмів роботи цих движків.

Якщо при розробці застосунку використовуються цінні алгоритми розробника, то такий код рекомендується захищати за допомогою ручної або автоматичної обфускації коду.

3.1.1 Специфіка розробки мобільного застосунку

При розробці застосунку рекомендується дотримуватись сформованих на основі проведеного дослідження правил.

Перший перелік правил стосується використання користувацького коду захисту (пасскоду):

- При використанні захисту пристрою паролем (ПІН-кодом, біометрією) при переході застосунку в режим очікування (при згортанні виконуваної програми) необхідно перекривати екран виконання застосунку паролем задля виключення випадку компрометації персональних даних при втраті пристрою, що не встиг заблокуватись;
- Кожен користувач повинен обмежуватися у спробах вводу паролю до застосунку з подальшим виходом з облікового запису або блокуванням застосунку в цілому після послідовності невдалого вводу;
- Формування цифрового паролю до застосунку повинно бути патернізованим та містити не менше 6 цифр.

Наступною ланкою безпеки є захищеність клієнт-серверного застосунку:

- При використанні серверної аутентифікації та авторизації необхідно використовувати сесійний механізм. Це дозволяє виключати можливість відкладеного доступу до застосунку, якщо користувач не здійснив вихід із застосунку. Слід зазначити, що термін дії сесії і її ідентифікатор відносяться до КВД, з усіма наслідками, що випливають звідси. Одним з вдалих прикладів реалізації подібного механізму є отримання абсолютного значення часу з сервера після проходження процедури авторизації користувача (дата і час повинні показувати, коли саме сесія стане неактивною). Дату і час

закінчення дії сесії не слід генерувати на пристрої, це знижує безпеку і гнучкість програми;

- Клієнт-серверний застосунок не повинен здійснювати зміну КВД в локальному режимі. Будь-яка дія, що вимагає зміни КВД, має проходити синхронно з сервером. Виняток з цього правила становить лише код входу користувача, що задається особисто користувачем і збережений в захищеному локальному сховищі.

При оперуванні важливими для роботи програми датами, на кшталт часу знищення сесії, не слід опиратися на відносну час. В силу наявності потенційно високих затримок у передачі даних по мережі від мобільного застосунку до сервера і назад, подібний спосіб синхронізації буде володіти надто великою похибкою. Крім того, атакуючий (або просто недобросовісний користувач) може просто змінити локальний пояс на пристрої, порушивши таким чином логіку роботи обмежувальних механізмів застосунку. Завжди потрібно передавати тільки абсолютне значення часу.

Абсолютні значення слід передавати з застосуванням універсальних способів обміну такою інформацією, без прив'язки до часового поясу конкретного користувача пристрою. Оптимальним варіантом є поведінка застосунку, при якому дані відображаються користувачу в його локальному часовому поясі, але їх зберігання і передача здійснюється в форматі, що не є прив'язаним до тайм-зони. Відповідними форматами для дат і часу є або універсальний UNIX timestamp, що зберігається в змінній 64-бітного цілого знакового типу (UNIX timestamp - це кількість секунд, що минуло з 1 січня 1970 року) або рядок в повному форматі ISO-8601 з нульовою тайм-зоною. Рекомендується використовувати саме UNIX timestamp, він дозволяє уникнути потенційних помилок і проблем з конвертацією рядків в дату і назад на різних мобільних платформах.

З додаткових рекомендацій слід додати наступне:

- Застосунок не повинен відображати приватну інформацію користувача великими, яскравими, добре зчитуваними шрифтами, без явної на те необхідності і без окремого запиту користувача, щоб виключити можливість читання цих даних випадковим чином з екрану пристрою;
- Не варто використовувати бібліотеки з відкритим вихідним кодом для захисту приватних даних користувача. Виняток становлять бібліотеки, перевірені часом і ті, що використовуються в великих проектах корпорацій (наприклад, вбудоване шифрування у відкритому движку бази даних Realm). Штатних механізмів захисту операційної системи і загальнодоступних перевірених криптографічних алгоритмів в переважній більшості випадків достатньо.
- Абсолютно неприпустимо використовувати криптографічні бібліотеки з закритим вихідним кодом. У таких рішеннях відсутня можливість перевірки ефективності бібліотеки, а також підтвердити якість;
- У кінцевих збірках застосунків необхідно вимикати логування даних в системну консоль і незахищені файли. Специфічні логи для розробників можуть бути присутніми, але бажано в зашифрованому вигляді, щоб уникнути доступу третіх осіб до закритої службової інформації, яка може бути присутня в логах.

Якщо розроблюваний застосунок буде мати справу з документами, тоді цей фактор накладає додаткову необхідність в забезпеченні вірної захищеності документів.

Одним із таких напрямків є документообіг в банківський установі. Стара система ведення юридичних справ клієнтів є дуже непродуктивною, прив'язаною до паперу та обмеженою в плані внесення будь яких змін до документів в час стрімкого руху розвитку, постійних змін правил та законів, стандартів та зразків ведення документації.

Саме тому проведення діджиталізації клієнтського документообігу є досить важливим витком розвитку як банківської, так і інших систем документообігу старих зразків. Така процедура передбачає повне переведення всього документообігу у електронний варіант з можливістю електронного підпису документів та їх шифрування (або належним безпечним зберіганням) на серверах документів.

Електронний цифровий підпис — вид електронного підпису, отриманого за результатом криптографічного перетворення набору електронних даних, який додається до цього набору або відповідно з ним поєднується і дає змогу підтвердити його цілісність та ідентифікувати підписанта. Електронний цифровий підпис накладається за допомогою особистого ключа та перевіряється за допомогою відкритого ключа.

Одним із елементів обов'язкового реквізиту є електронний підпис, який використовується для ідентифікації автора або підписанта електронного документа іншими суб'єктами електронного документообігу. Оригіналом електронного документа вважається електронний примірник з електронним цифровим підписом автора.

Електронний цифровий підпис призначений для використання фізичними та юридичними особами - суб'єктами електронного документообігу:

- Для ідентифікації підписанта;
- Для підтвердження цілісності даних в електронній формі.

ЕЦП як спосіб ідентифікації підписанта електронного документу, дозволяє однозначно визначати походження інформації (джерело інформації), що міститься у документі. Завдяки цьому ЕЦП є також надійним засобом розмежування відповідальності за інформаційну діяльність у суспільстві, зокрема, відповідальності за дезінформування.

Накладання ЕЦП завершує утворення електронного документа, надаючи йому юридичної сили.

Згідно закону України «Про електронні документи та електронний документообіг» юридична сила електронного документа з нанесеними одним або множинними ЕЦП та допустимість такого документа як доказу не може заперечуватися виключно на підставі того, що він має електронну форму.

Відповідно до «Положення про організацію заходів із забезпечення інформаційної безпеки в банківській системі України» для електронного підпису дозволяється використовувати наступні алгоритми:

- Алгоритм цифрового підпису (далі – алгоритм DSA) для цифрових підписів;
- Алгоритм цифрового підпису на еліптичних кривих (алгоритм ECDSA) для цифрових підписів;
- Алгоритм Ривест – Шаміра – Адлемана (далі – алгоритм RSA) для цифрових підписів;
- Алгоритм цифрового підпису (ДСТУ 4145-2002 “Інформаційні технології. Криптографічний захист інформації. Цифровий підпис, що ґрунтується на еліптичних кривих. Формування та перевіряння”, затверджений наказом Державного комітету України з питань технічного регулювання та споживчої політики від 28 грудня 2002 року № 31) (далі – алгоритм ДСТУ 4145-2002) для цифрових підписів.

Для реалізації електронного підпису документів пропонується використовувати відкриту бібліотеку роботи з PDF файлами під назвою «Apache PDFBox®». В обраній бібліотеці реалізовано механізм ЕЦП за допомогою алгоритму RSA, що задовольняє умовам «Положення про організацію заходів із забезпечення інформаційної безпеки в банківській системі України». Використовуючи відкриту бібліотеку розробники мають доступ до її виконуваних модулів. Таким чином є можливість пересвідчитися у правильності та захищеності

механізмів, що використовує бібліотека, та також переконатися у відсутності прихованих «підводних каменів».

Алгоритм підпису документу полягає в наступних кроках:

1. Отримується документ
2. Створюється цифровий підпис *s* за допомогою власного секретного ключа *{d,n}* за формулою 2.1:

$$s = S_A(m) = m^d \ mod \ n \qquad (3.1)$$

де s – підписаний файл;

m – не підписаний файл;

d – секретний персональний ключ;

n – просте число відоме число.

Механізм підписування документів базується на документообігу файлів формату PDF. Підписання файлів реалізується в автоматичному вигляді за допомогою персональних закритих ключів, які зберігаються на електронних носіях користувачів застосунку,.

Перевірка підпису документа вже реалізована в самій програмі перегляду PDF файлів, тому ця функція покладається на програмне забезпечення, що вже встановлено на пристроях.

Наочна схема роботи підпису повідомлень (документів) зображена на рис. 3.1:

Рисунок 3.1 – Схема роботи електронного цифрового підпису

3.1.2 Специфіка розробки під платформи iOS та Android

При розробці під операційну систему iOS специфіка полягає у виборі сховища даних застосунку:

- NSUserDefaults – захищеність цього типу сховища відсутня і воно використовується виключно для збереження не приватних даних застосунку задля його функціонування, використовується для запису налаштувань інтерфейсу;
- Бінарні файли (NSKeyedArchiver) – за замовчуванням при використанні атрибуту NSFileProtectionComplete за ключем NSFileProtectionKey присутня. Рівень захищеності залежить виключно від алгоритмів шифрування, що приміняться до таких файлів. При виконанні вище зазначених умов файли є захищеними поки пристрій заблоковано чи знаходиться в стані завантаження, тому це вважається лише тимчасовою мірою захисту і не рекомендується до використання із приватними даними;
- Бази даних – повністю залежне від розробника сховище даних. БД підтримують шифрування і вже розробником обирається сторона швидкодії застосунку чи його більш стійкого алгоритму шифрування БД;

- Ланцюг ключів (Keychain) – найбільш захищене сховище даних, якщо не враховувати пристроїв з джейлбрейком. При використанні keychain варто обережно налаштовувати синхронізацію з хмарою, адже елементи ланцюга можуть автоматично зберегтися в хмарі, що піддає ризику приватність даних. Рекомендовано використовувати для збереження КВД, попередньо зашифровуючи дані користувача.

Для Android рекомендовано використовувати наступні налаштування та правила розробки застосунку:

- Запит на дозволи для застосунку повинен повідомляти про всі функції та дані пристрою, які застосунок буде використовувати в рамках своєї роботи. Також необхідно жорстко обмежити дозволи для застосунку по мірі їх необхідності – запитувати дозвіл виключно через необхідність у його використанні. Сам запит для версій Android SDK від 23 варто проводити через Compatibility Permissions System;
- При використанні паролю рекомендовано графічний пароль та 6-значний (мінімум) як допоміжний;
- При використанні HTTP-клієнту примусово повинно бути налаштовано HTTPS з'єднання;
- При виході в використання в застосунку повинно вимкнути всі debug функції задля уникнення підключення сторонніми програмами для аналізу відлагоджувальної інформації застосунку;
- На екранах, що можуть відображати приватну інформацію варто заборонити створенні знімків екрану;
- На запит будь-якої приватної інформації необхідно запитувати ключ авторизації в застосунок (якщо такий присутній);

- Відкриті компоненти застосунку (Service, Content Provider) повинні бути помічені позначкою exported=false в файлі маніфесту застосунку задля унеможливлювання доступу до цих компонентів.

З вибором сховища для Android застосунку існує чотири варіанти. Незахищеним сховищем є Shared Preferences, що повинно використовуватись виключно для зберігання загальних відкритих користувацьких налаштувань.

Бази даних (SQLite), як і по аналогії з iOS, є захищеними залежно від вибору розробником алгоритму шифрування цих баз даних. В якості ключа шифрування варто використовувати код захисту застосунку, що в свою чергу повинен бути зашифрований та поміщений в Account Manager.

Account Manager в свою чергу є придатним для збереження КВД, але попередньо шифруючи такі дані перед переміщенням до цього сховища.

Починаючи з версії API 18 на зміну Account Manager прийшов KeyStore, який зараз рекомендується використовувати замість вищезазначеного.

3.2 Захист серверної частини застосунку

За останніми тенденціями є досить популярним використання хмарних середовищ для розміщення власних обчислювальних програм та серверів. Саме такий підхід пропонується використовувати при розробці застосунку, адже хмарні сервіси в наш час є досить детальні в налаштуваннях в залежності від специфіки їх використання та також пропонують високий рівень спостережності, керованості та підтримки.

Набагато вигідніше використовувати готові рішення хмарних послуг, чим орендувати обчислювальні потужності на хостингах або ж налаштовувати власні потужності, адже головною проблемою при «ручному» підході є швидкість реагування на різноманітні новоявлені загрози та атаки.

При організації системи безпеки виникає ряд загроз, які необхідно враховувати, та які приносять велику загрозу в систему.

Приводиться невтішна статистика розвитку таких загроз:

- Кожного дня фіксується близько 400 000 нових шкідливих об'єктів;
- Набирають популярності нові види шкідливих програм, до яких відносять Ransomware, Scareware і багато інших, що націлені в основному на банківський сектор;
- Зазвичай вектор атаки направлений на публічні хмарні сервіси, інфраструктури SaaS- та IaaS- провайдерів, на мобільні пристрої;
- Збільшення кількості використання зловмисниками так званих ботнетів та інших варіантів автоматизації атак;
- Відроджуються старі атаки типу Heartbleed, ShellShock і Poodle;
- Вектор атак значно розширюється, що несе з собою колосальні фінансові збитки.

Провайдерам хмарних послуг необхідно знати, що вони є першою потенційною ціллю зловмисників, тому повинні постійно слідкувати за ситуацією на ринку, контролювати мережевий трафік, активність додатків системи, аналізувати переміщення даних на всіх рівнях хмарної інфраструктури, при цьому мінімально впливаючи на клієнтів.

Також досить вагомим фактором безпеки є швидкість реагування. В порівнянні з традиційними IT-середовищами, в хмарному середовищі цей показник вважається критичним.

3.2.1 Визначення рівнів інфраструктури, потребуючих захисту

Першим кроком в реалізації хмарної безпеки є ідентифікація рівнів середовища, що потребують захисту. Наведена нижче схема, зображена на Рисунку 3.1, ілюструє узагальнені рівні хмарного оточення.

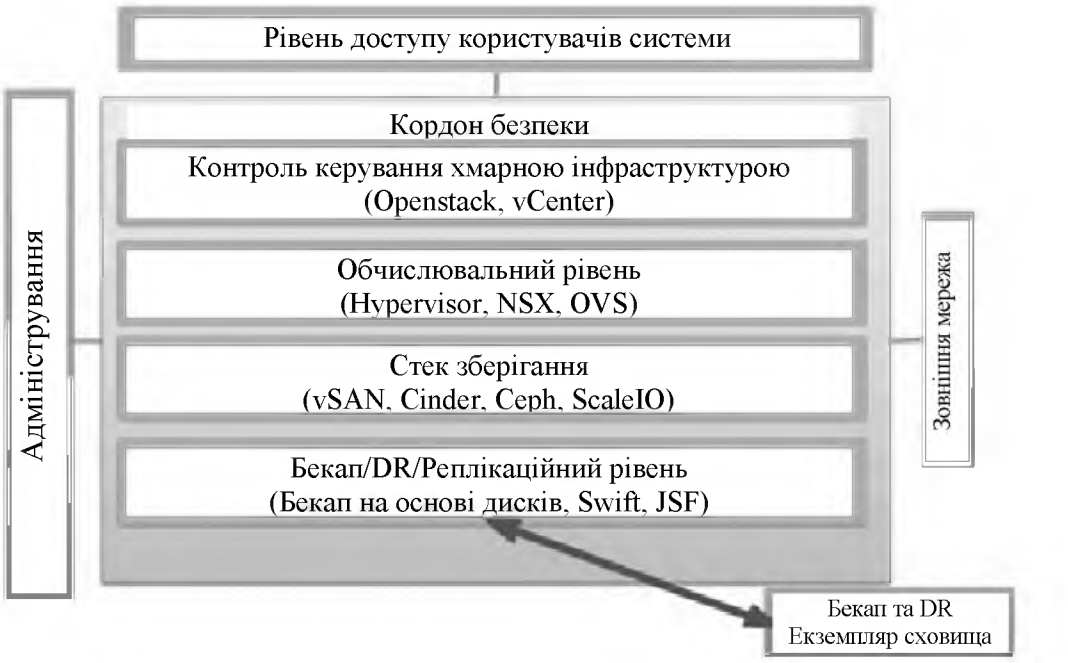

Рисунок 3.2 – Схема рівнів безпеки хмарного сервісу

Ці рівні в більшості своїй є багатошаровою складовою, вміст і компоненти якої на кожному з рівнів відрізняються один від одного. При цьому деякі наведені приклади властиві тільки для хмарних середовищ:

- Гіпервізор;
- Віртуальна мережа;
- Віртуальне сховище;
- Інструменти і автоматизація управління;

- Компоненти SDN (програмно конфігурується мережу);
- Портал самообслуговування.

3.2.2 Вибір методології захисту і інструментів

Наступний крок зводиться до визначення методів захисту для кожного рівня. В якості загальної рекомендації пропонується розглядати екземпляр хмари як зону безпеки, в якій визначено межі безпеки з присутніми там проксі і файрвол (як в віртуальних, так і в фізичних середовищах). Після того як визначені межі безпеки, варто звернути увагу на методи, спрямовані на виконання моніторингу, корекцію підозрілої активності і захист від шкідливих програм.

Важливо відзначити, що постачальник хмарних послуг не може використовувати інструменти або механізми доступу до віртуальних машин замовника. Приклад тому - захист від шкідливих програм. Такий захист на рівні гіпервізора є вкрай необхідним, але в дійсності зачіпає віртуальні машини клієнта, що порушує кордон між провайдером і замовником. У цьому допомагають інструменти та конфігураційні опції, перераховані нижче.

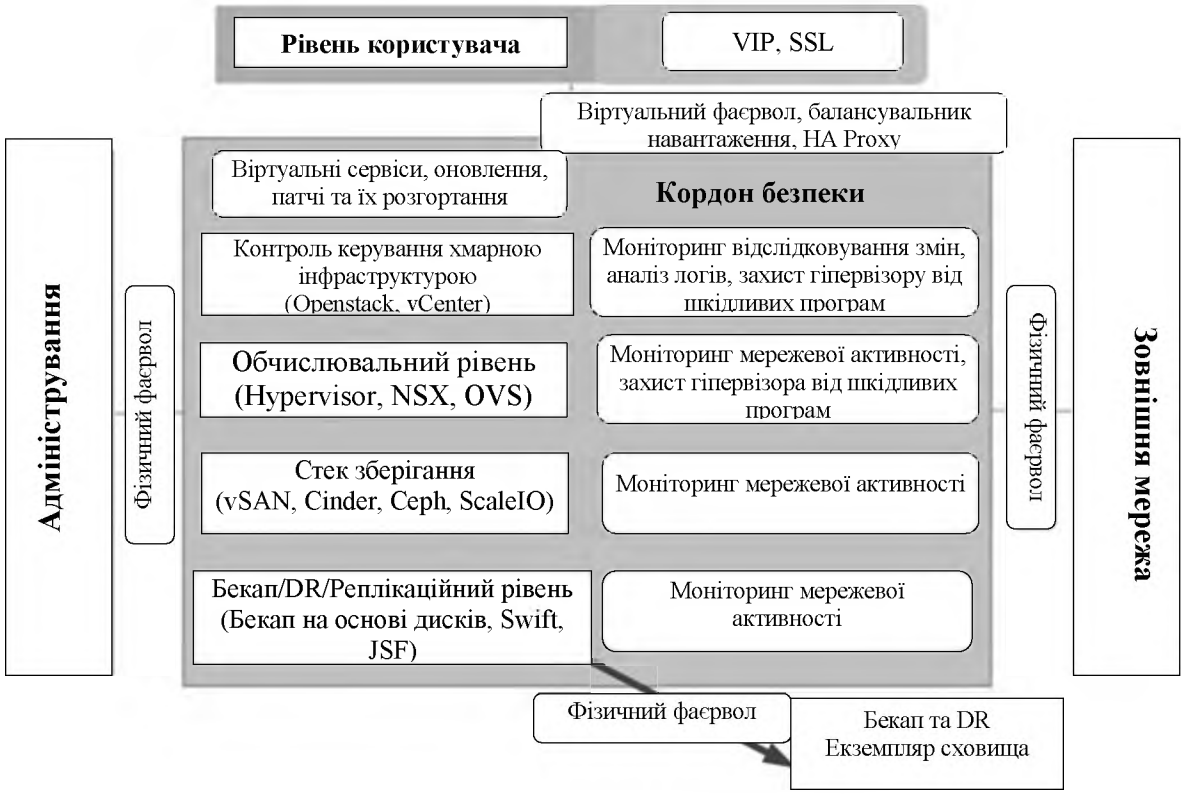

Рисунок 3.3 – Схема захисту хмарного сервісу

- VIP: надання віртуальних IP-адрес дозволяє розділяти мережі на внутрішні і зовнішні.
- SSL: забезпечує шифрування HTTP-трафіку.
- Периметри безпеки (міжмережеві екрани, засоби балансування навантаження, проксі): визначають межі зон безпеки і дозволяють управляти потоком трафіку, що проходить між внутрішніми і зовнішніми мережами. Міжмережеві екрани, балансувальник навантаження і проксі-сервери можуть бути представлені у вигляді апаратних або програмних рішень. Останні часто встановлюються на віртуальні машини в хмарі провайдера.
- Віртуальні сервіси: «аплайнси» безпеки або віртуальні машини, що відповідають за оновлення, патчі і сервіси всередині зон безпеки.
- Моніторинг мережевої активності / IDS: моніторинг проходить трафіку з метою виявлення вторгнень. У хмарному оточенні повинні бути присутніми

спеціалізовані інструменти, що дозволяють аналізувати мережевий трафік віртуальних машин.

- Відстеження змін файлів: стосується важливих конфігураційних файлів керованої інфраструктури, включаючи рівень гіпервізору.
- Відстеження логів і їх аналіз: централізоване відстеження подій за допомогою лог-файлів (наприклад, Syslog, журнали компонентів керованої хмари) і аналіз подій для оцінки тенденцій і виявлення шкідливої активності.
- Захист від шкідливих програм на рівні гіпервізора: спеціалізоване програмне забезпечення (є досить велика кількість на ринку) для виявлення і видалення шкідливого ПЗ з гіпервізора і фізичних пристроїв.

3.2.3 Стратегія безпеки

Важливим аспектом роботи хмарного сервісу є пророблений план підтримки безпеки інфраструктури. Представлено наступний план, якому необхідно слідувати для забезпечення захищеності:

- Регулярне оновлення програмного забезпечення (програмне забезпечення вендорів, хмарний гіпервізор, компоненти систем безпеки - все це повинно оновлюватися постійно). Будь-які зміни необхідно фіксувати і аналізувати, а всі оновлення застосовувати належним чином;
- Регулярне внесення виправлень: застосування патчів і усунення помилок. Цей пункт є високопріоритетним для хмарної середовища;
- Централізована активність і моніторинг компонентів безпеки: наявність згуртованої команди фахівців, використання інструментів моніторингу, виконання оцінки існуючої активності, застосування алертів безпеки віртуального оточення. Все перераховане дозволяє швидко розпізнавати виникає активність і в разі потреби вносити коригування;

- Завдання за розкладом і проведення тестів на визначення вразливостей: необхідність завжди бути напоготові. Інфраструктура, яка здається надійною сьогодні, може піддатися загрозам і атакам нового формату завтра. Тому регулярно виконуються тести на визначення вразливостей і тести після застосованих оновлень допоможуть мінімізувати ризики і зберегти інфраструктуру в безпеці;
- Коригування процедур управління і відстеження змін: одним з кроків визначення «шкідливих» оновлень є відстеження змін і порівняння їх з результатами сканування змінених файлів. Такі процедури можуть допомогти в загальному вирішенні проблем та способи їх усунення положень безпеки;
- Правильне управління середовищем в цілому вимагає, щоб процеси і процедури, особливо що стосуються безпеки, регулярно переглядалися і коригувалися, а в разі необхідності вносилися зміни.

3.2.4 Додаткові інструменти захисту

Важливою технологією, що використовується в запропонованій системі безпеки хмарних IaaS-сервісів, вважається технологія детального дослідження пакетів DPI (Deep Packet Inspection).

Перш за все перевагою DPI є можливість реалізації даної технології в вигляді програмному комплексу або в вигляді фізичного пристрою, що здатен виконувати аналіз трафіку 6-ти рівнів моделі OSI та класифікувати його, що зображено в Таблиці 3.1.

Таблиця 3.1 – Модель OSI та DPI

Рівень	Модель OSI	Об’єкт керування	Можливість обробки DPI
1	Прикладний	Дані	Так
2	Представлення	Дані	Так
3	Сеансовий	Дані	Так
4	Транспортний	Блоки	Так
5	Мережевий	Пакети	Так
6	Канальний	Кадри	Так
7	Фізичний	Біти	-

Таким чином використання DPI забезпечує:

- Розподіл каналу між застосунками (QOS), користуючись попередньо прописаними інструкціями розробника;
- Ведення статистики по використовуваним застосункам та API цих застосунків при сценарії, коли сторонні сервіси можуть використовувати розроблюваний застосунок, приклад якої зображено на Рисунку 3.3;
- Захист від атак базуючись на аналізі трафіку: TCP/UDP Flood;
- Залучення розширення VAS (Value Added Services) для детальнішого аналізу.

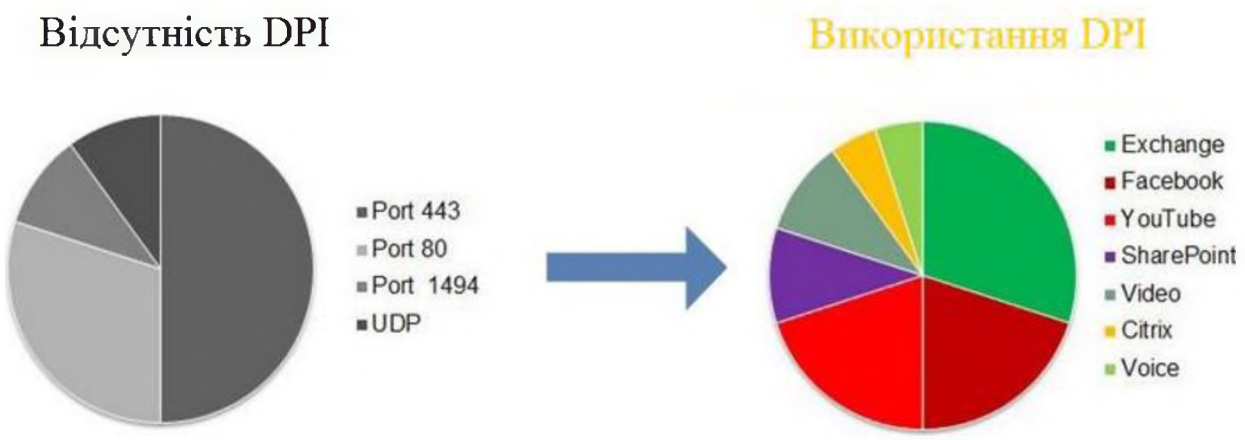

Рисунок 3.4 – Приклад використання DPI аналізу

Наступним важливим інструментом є резервне копіювання в хмару. В розроблюваній архітектурі використовується технологія Veeam Cloud Connect, що забезпечує три основні правила резервного копіювання «3-2-1»:

- наявність трьох резервних копій;
- для зберігання копій використовувати два носія;
- один екземпляр повинен зберігатись віддалено.

Технологія Veeam Cloud Connect дозволяє забезпечувати:

- Надійне зберігання резервних копій на віддалених майданчиках постачальників хмарних послуг;
- Комплексний контроль з гарантованою можливістю відновлення даних, що зберігаються на віддалених майданчиках, безпосередньо з консолі резервного копіювання;
- Можливість використання завдань архівування резервних копій за допомогою вбудованої акселерації WAN;

- Можливість використання постійно інкрементального резервного копіювання та політик довгострокового зберігання.

На вибір пропонується три продукти, що підтримують таку технологію: Veeam Availability Suite™ v9.5, Veeam Backup & Replication™ v9.5, Veeam Backup Essentials™ v9.5.

Останнім додатковим інструментом забезпечення безпеки є продукт шифрування даних в хмарному середовищі. Вибором для забезпечення такої функції став продукт SecureCloud компанії Trend Micro, компанії зі світовим ім'ям, основним профілем діяльності якої є корпоративна безпека.

Модель використання рішення передбачає, що диски віртуальних машин зашифровуються з використанням ключів шифрування, які зберігаються в системі SecureCloud. Через систему SecureCloud ініціюються процеси початкового шифрування або розшифрування захищених дисків. При спробі доступу до даних відбувається звернення до системи SecureCloud, в результаті якого, в залежності від системних політик, відбувається або автоматична разова видача ключа шифрування для розшифровки даних (наприклад, для завантаження застосунку), або видача ключа тільки після схвалення адміністратором.

3.3 Організаційний аспект безпеки використання мобільних застосунків

Для забезпечення належного рівня захищеності необхідно розробити відповідну чітку політику безпеки використання мобільного застосунку. Оскільки головною ціллю дипломної роботи є створення більш універсальної системи захисту, тому було обрано обов'язкові пункти, які політика безпеки повинна буде містити. Нижче зазначений рекомендаційний перелік складових політики, що

забезпечує базовий необхідний рівень захищеності, не обмежує спеціалізацію застосування такої політики безпеки, тобто вміщає універсальні елементи.

Попередньо необхідно виконати наступні кроки:

1. Створити підрозділ та/або робочу групу з мобільних технологій (з метою координації робіт, розробки політики мобільної безпеки і організації взаємодії з іншими компонентами інформаційної інфраструктури організації).
2. Визначити стандарти використання мобільного застосунку.
3. Визначити авторизованих користувачів та їх групи, дослідити потенційну категорії користувачів.
4. Класифікувати дані. Слід визначити категорії критичності ресурсів і даних, до яких дозволений мобільний доступ.
5. Встановити повноваження доступу.
6. Розробити реалістичну політику безпеки.

Після виконання зазначених пунктів варто приділити увагу власне змісту політики безпеки. Отже, до політики безпеки мобільного застосунку необхідно включити наступні пункти:

- Перелік користувачів, що мають мобільний доступ до ресурсів застосунку;
- Перелік платформ або моделей пристроїв, які можуть використовувати застосунок;
- Дії, які необхідно виконати при втраті пристрою з чутливими даними застосунку на втраченому пристрою;
- Використання паролів доступу до ресурсів застосунку;
- Зобов'язання до того, щоб всі використовувані паролі були складними, пропрацювати можливий варіант використання патернів створення паролів,

що вміщатимуть латиницю, символи розмітки, цифрові символи та також символи різних регістрів;

- Необхідність використовувати шифрування даних застосунку;
- Необхідність контролювати використання застосунку, базуючись на інформації про підключення пристрою до WiFi-мереж або мережевого Інтернет підключення, та на основі політик доступу і даних про місце розташування пристрою, блокувати деякі важливі та небезпечні для застосунку функції (критичний варіант контролю захищеності даних). До таких функцій відноситься копіювання та вставка даних, служби визначення місцезнаходження, електронна пошта та застосунки для обміну повідомленнями, використання камери та мікрофону.

З точки зору роботи з користувачами такого мобільного застосунку необхідно підготувати інструкцію до використання мобільних пристроїв в ролі, що запропонована даною концепцією. Тому наступним кроком організації роботи систем безпеки є опрацювання необхідної інформації про інформаційну безпеку застосунку. Таким чином правильною практикою є включення необхідних правил ознайомлення та постійного контролю за якістю знань щодо інформаційної безпеки у користувачів за допомогою наступних заходів:

- вступний інструктаж з питань забезпечення інформаційної безпеки в застосунку;
- тематичні пам'ятки, календарі, флеш-ролики тощо, які нагадують користувачу про важливість питань інформаційної безпеки;
- Інтегрування в систему заохочень у вигляді нагороди за виконання тих чи інших мір захисту персональних даних.

3.4 Архітектура системи захисту мобільного застосунку

В кінцевому випадку необхідно зібрати всі рекомендації, правила, техніки та технології для забезпечення максимальної захищеності застосунку.

В цілому виділяючи застосунок як окрему систему варто дотримуватись передбачених правил та технік безпеки розробки мобільного застосунку, використовувати графічні рішення для запобігання втрати приватної інформації шляхом випадкового доступу до легко розглядаємої інформації, реалізувати ряд інструкцій користувача, дотримання яких передбачає нагороду та заохочення.

Наступною частиною такої архітектури є серверна частина, яка використовується для обчислень мобільного застосунку. Необхідно переносити якомога більше функціоналу саме на серверну частину застосунку, адже вона є більш захищеною, правила захищеності є більш доступні для налаштувань, а це гарантує швидку реакцію на різні новостворені види загроз. Як готові рішення передбачено використання DPI (Allot NetEnforcer) для фільтрування запитів, що надходять, включення в систему реалізацій хмарних віртуальних машин, їх шифрування (SecureCloud Trend Micro), резервного копіювання (Veeam Backup & Replicatіon™ v9.5) та розподілу навантажень (vCloud VMware).

Заключним головним елементом такої архітектури є системи безпеки на мобільному пристрої. При розробці безпеки локального застосунку варто враховувати особливості інструментів забезпечення захищеності застосунків для обраної платформи, використовувати патернизовані паролі при доступу до приватних даних, що повинні знаходитись в зашифрованому вигляді в захищеному сховищі даних операційної системи.

Таким чином описана система представлена на Рисунку 3.5.

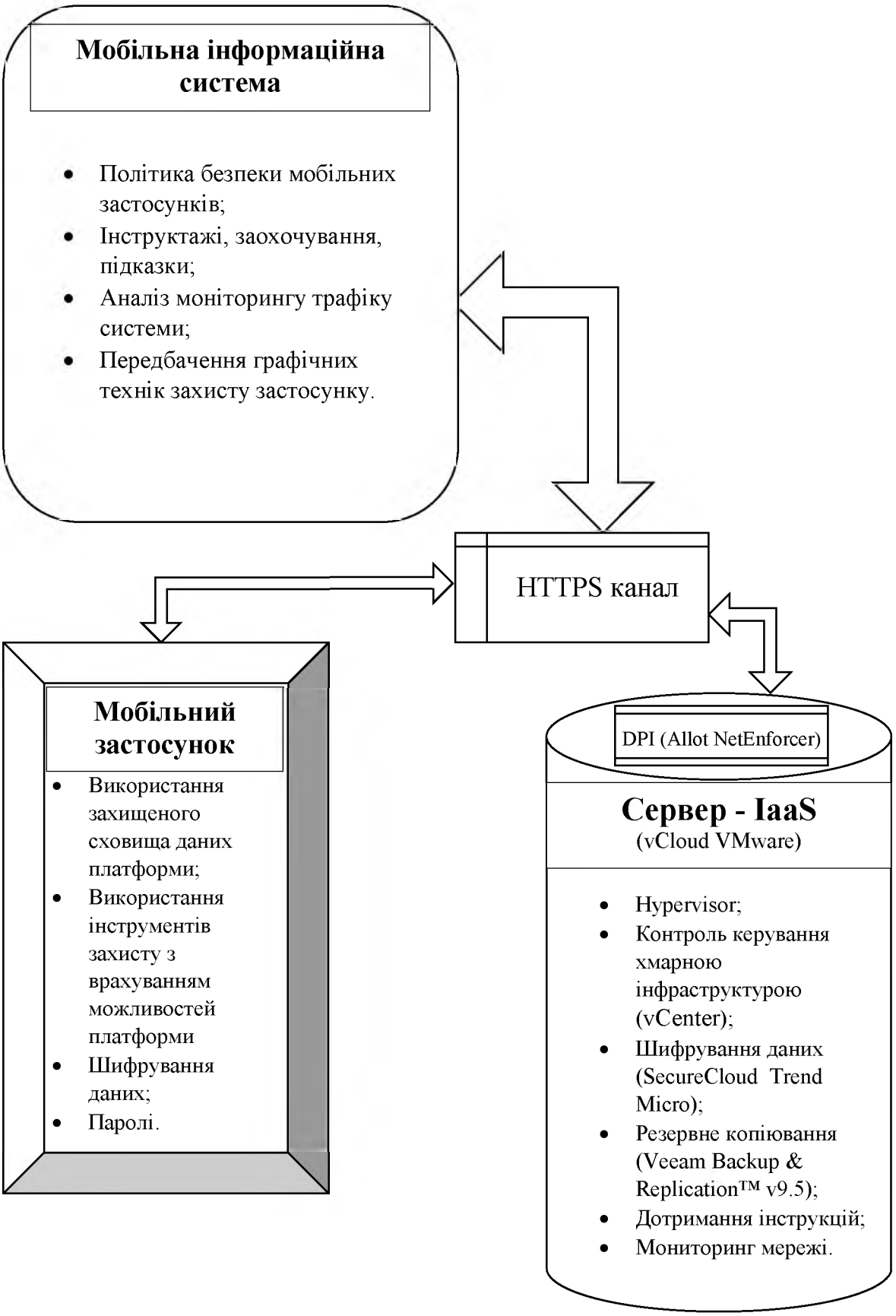

Рисунок 3.5 – Схема архітектури системи захисту мобільного застосунку

Висновки до розділу 3

В даному розділі було досліджено методи боротьби з основними загрозами мобільних платформ iOS та Android та також спільні для цих платформ механізми захисту застосунків. На основі вивченого ринку мобільних послуг, мобільних платформ та загроз їх безпеки було запропоновано архітектуру системи безпеки мобільного застосунку, що включає в себе технічні та організаційні рішення для максимальної організації захищеності застосунку та критично важливих користувацьких та системних даних, що циркулюють в застосунку.

4 РОЗРОБКА СТАРТАП ПРОЕКТУ

4.1 Опис ідеї продукту

Таблиця 4.1 – Опис ідеї стартап-проекту

Зміст ідеї	*Напрямки застосування*	*Вигоди для користувача*
Модернізація мобільного застосунку з метою підвищення ступеню його захищеності. Аудит безпеки мобільних застосунків, попередження атак мобільних застосунків, надання рекомендацій щодо забезпечення захищеності мобільних застосунків	1. Великі та середні компанії-розробники мобільного ринку застосунків	Забезпечення захищеності приватних даних користувачів та як наслідок привернення більшої аудиторії користувачів
	2. Компанії, що утримують маркети мобільних застосунків	Підтримка захищеності застосунків у власному магазині мобільних застосунків.

Таблиця 4.2 – Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№ п/п	*Техніко-економічні характеристики ідеї*	*(потенційні) товари/концепції конкурентів*		*W (слабка сторона)*	*N (нейтральна сторона)*	*S (сильна сторона)*
		Стартап проект	*Koloro*			
1	2	3	4	5	6	7

Продовження таблиці 4.2.

1	2	3	4	5	6	7
1	Технічна характеристика	Аналіз захищеності застосунку на основі запропонованої архітектури	Використання внутрішньої методики	Об’ємність робіт з дослідження та подальшої розробки	Схожий підхід до вирішення загроз	Універсальність рішення, базування підходу на системах з нульовим знанням
2	Економічна характеристика	Виконання робіт проводиться в залежності від продукту, об’єктивна вартість повного циклу коливається близько 10000 $	Проводиться повне обстеження продукту, виконуються необхідні роботи у повному обсязі. Вартість від 15000$	Об’ємність робіт з дослідження та подальшої розробки	немає	Економічність рішення, варіативність наданих послуг, вигідність рішення

Продовження Таблиці 4.2

1	2	3	4	5	6	7
3	Технологічна характеристика	Використання хмарних технологій при розробці серверної частини	Відсутність можливості реалізації серверної частини	Додаткова складність використання хмарних технологій	Немає	Більший рівень захищеності при використанні хмарних технологій
4	Характеристика надійності	Забезпечення захищеності застосунку з боку хмари та всередині застосунку	Розробка системи безпеки локального застосунку	Додаткова складність використання хмарних технологій	Немає	Більший рівень захищеності при використанні хмарних технологій
5	Характеристика ефективності	Використання підходу з врахуванням користувацьких вподобань та можливостей	Розробка виключно з боку кодової частини застосунку	Немає	Немає	Врахування користувацького досвіду, більша лояльність серед користувачів

Продовження Таблиці 4.2

1	2	3	4	5	6	7
6	Характеристика безпеки	Використання хмарних сервісів, врахування загроз встановленому застосунку	Врахування тільки загроз, пов'язаних з локальним застосунком	Додаткові витрати ресурсів на забезпечення хмарних сервісів	Немає	Ширший підхід до розробки системи захисту забезпечує більшу якість та безпечність застосунку

4.2 Технологічний аудит ідеї проекту

Таблиця 4.3 – Технологічна здійсненність ідеї проекту

№ п/п	*Ідея проекту*	*Технології її реалізації*	*Наявність технологій*	*Доступність технологій*
1	2	3	4	5
1	Розробка системи захисту базуючись на особливостях платформи	Інструменти та особливості платформ iOS та Android	Необхідність використання інструментів розробки застосунків	Технологія доступна

Продовження Таблиці 4.3

1	2	3	4	5
2	Розробка інструкцій та правил з безпеки застосунку	Загальноприйняті та логічні аспекти захисту мобільного застосунку	Технологія наявна	Технологія доступна
3	Використання хмарних технологій з боку серверної частини	Технології vCloud VMware, Veeam та SecureCloud	Оформлення необхідних сервісів та їх налаштування	Технологія доступна
Обрана технологія реалізації ідеї проекту: всі перераховані технології наявні та готові до використання, тому реалізація полягає у поступовому об'єднанні перерахованих засобах та інструментах.				

4.3 Аналіз ринкових можливостей запуску стартап-проекту

Таблиця 4.4 – Попередня характеристика потенційного ринку стартап проекту

№ п/п	*Показники стану ринку (найменування)*	*Характеристика*
1	Кількість головних гравців, од	5 од
2	Загальний обсяг продаж, грн/ум.од	1 млн ум.од
3	Динаміка ринку (якісна оцінка)	Зростає

Продовження Таблиці 4.4

1	2	3
5	Специфічні вимоги до стандартизації та сертифікації	Отримання спеціалізованих сертифікатів для підвищення статусності рішень захисту
6	Середня норма рентабельності в галузі (або по ринку), %	Не менше 130%

За загальними оцінками ринок забезпечення захисту мобільних застосунків є досить відкритим та зростаючим, конкуренція перебуває на низькому рівні, тому є підстави вважати цей ринок привабливим для інвестицій.

Таблиця 4.5 – Характеристика потенційних клієнтів стартап-проекту

№ п/п	*Потреба, що формує ринок*	*Цільова аудиторія (цільові сегменти ринку)*	*Відмінності у поведінці різних потенційних цільових груп клієнтів*	*Вимоги споживачів до товару*
1	Незахищеність мобільних застосунків, їх велика розповсюдженість, великий вміст приватної інформації в мобільних застосунках	- Компанії-розробники мобільних застосунків; - Компанії-утримувачі мобільних маркетів та платформ.	Для наведених категорій відмінність полягає у обсягах забезпечення захисту застосунків	- Висока захищеність кінцевого продукту; - Легка підтримка продукту після проведених робіт

Таблиця 4.6 – Фактори загроз

№ п/п	*Фактор*	*Зміст загрози*	*Можлива реакція компанії*
1	Потреба захищеності мобільних застосунків	Коливання потреби в розробці систем захищеності мобільних застосунків	Пошук маркетингових ходів з метою популяризації захищеності продуктів
2	Цінова конкуренція	Зміни в цінових політиках компаній-конкурентів	Пошук шляхів здешевлення послуг, що надаються
3	Зміни державного рівня	Зміна податків, ситуації в країні-утримувачі	Зміна ринку виробництва

Таблиця 4.7 – Фактори можливостей

№ п/п	*Фактор*	*Зміст можливості*	*Можлива реакція компанії*
1	2	3	4
1	Розвиток технологічних рішень у сфері безпеки мобільних застосунків	Покращення якості та можливостей захисту мобільних застосунків, надійності таких рішень та їх універсальності та стійкості до загроз	Оновлення технічної бази послуг, що надаються, проведення відповідних навчальних робіт працівників, опрацювання нових технологій
2	Стрімкий ріст кількості мобільних застосунків	Збільшення потреби в організації захищеності мобільних застосунків	Нарощування потужності з можливості надання послуг

Продовження Таблиці 4.7

1	2	3	4
3	Стабілізація ситуації в країні	Зменшення рівня інфляції, стабілізація курсу валют, іміджу країни як члену ринку послуг	Поява нових клієнтів, можливість стабільної роботи на міжнародному ринку

Таблиця 4.8 – Ступеневий аналіз конкуренції на ринку

1	2	3
Особливості конкурентного середовища	*В чому проявляється дана характеристика*	*Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)*
1. Олігополістична конкуренція	Наявність на ринку невеликої кількості продавців послуг	Можливість увійти в сферу за рахунок новизни рішень
2. За рівнем конкурентної боротьби - міжнародний	Універсальність реалізації послуг закордоном	Можливість значного розширення ринку збуту послуг
3. За галузевою ознакою - міжгалузева	Послуги є широко застосовуваними в різних сферах	Наявність широкого ринку збуту послуг
4. Конкуренція за видами товарів: - товарно-видова	Послуги, що виконуються для потреб клієнтів, але відрізняються за технологією роботи	Комплексний підхід до надання послуг

Продовження Таблиці 4.8

1	2	3
5. За характером конкурентних переваг - цінова	Відмінність вартості послуг	Більша цінність наданих послуг за одиницю грошей
6. За інтенсивністю - марочна	Визначення підходу до надання сервісів	Розширення та покращення спектру послуг, що надаються

Таблиця 4.9 – Аналіз конкуренції в галузі за М. Портером

	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
Складові аналізу	- Koloro - Guardant - Appomart	Розмір грошових вкладень в розвиток;	Поставки хмарних послуг	Контроль якості, розміри запитів на послуги	Ціна послуг та змінні витрати
Висновки	Конкуренція на ринку присутня, але через обмежену кількість послуг вважається слабкою	Можливість виходу на ринок висока, при пропозиції високоякісних послуг	Постачальники напряму не впливають на ринок, адже містять в собі тільки частинну залежність	Клієнти напряму диктують умови ринку, адже виключно вони зацікавлені в отриманні захищених застосунків	Товари-замінники не поширені на ринку

Існує ймовірна можливість продуктивної роботи запропонованого продукту через невиражену конкуренцію та сильну залежність клієнтів від підходу до надання послуг, що пропонуються. Серед сильних сторін проекту виділяється інноваційне використання хмарних сервісів в архітектурі захисту, включення до заходів безпеки розробки правил та політик безпеки та фактичну відсутність аналогів запропонованому підходу на території України. Також стрімкий розвиток обраної сфери застосування з великою ймовірністю стимулюватиме розвиток проекту в майбутньому.

Таблиця 4.10 - Обґрунтування факторів конкурентоспроможності

№ п/п	*Фактор конкурентоспроможності*	*Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)*
1	Ціна та змінні витрати	Конкурентна вартість послуг та відносно виправдана вартість змінних витрат у вигляді плати за хмарні сервіси
2	Запропонований підхід до послуг	Унікальність підходу до розробки системи безпеки на ринку
3	Доступність ресурсів для якісних розробок послуг	Ресурси полягають виключно в підході до надання послуг та кваліфікованих кадрах
4	Гнучкість підходу до розробки послуг	Використовуваний підхід забезпечує максимальну гнучкість рішень та підлаштування під унікального клієнта

Таблиця 4.11 – Порівняльний аналіз сильних та слабких сторін

№ п/п	*Фактор конкурентоспроможності*	*Бали 1-20*	*Рейтинг товарів-конкурентів у порівнянні з SMSec*						
			–3	*–2*	*–1*	*0*	*+1*	*+2*	*+3*
1	2	3	4	5	6	7	8	9	10

Продовження Таблиці 4.11

1	2	3	4	5	6	7	8	9	10
1	Ціна та змінні витрати	18		+					
2	Запропонований підхід до послуг	19	+						
3	Доступність ресурсів для якісних розробок послуг	12				+			
4	Гнучкість підходу до розробки послуг	16			+				

Таблиця 4.12 – SWOT-аналіз стартап проекту

Сильні сторони: унікальність підходу до розробки системи безпеки, використання хмарних сервісів, гнучкість запропонованих послуг	*Слабкі сторони:* середньо забезпечена маркетингова складова проекту, необхідна постійна підтримка захищеності в плані конфігурування хмарних сервісів
Можливості: гнучкість налаштуванн послуг в залежності від клієнта, залучення потужної клієнтської бази, популяризація послуг забезпечення захисту мобільних застосунків сторонніми постачальниками	*Загрози:* цінова конкуренція, нестабільна ситуація в країні, невдала маркетингова кампанія

Таблиця 4.13 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	*Альтернатива (орієнтовний комплекс заходів) ринкової поведінки*	*Ймовірність отримання ресурсів*	*Строки реалізації*
1	2	3	4

Продовження Таблиці 4.13

1	2	3	4
1	Звуження спеціалізації використовуваних технологій	Висока ймовірність отримання ресурсів	1 рік
2	Надавання послуг базуючись на одній екосистемі хмарних послуг	Висока ймовірність отримання ресурсів та залучення підтримкою вендора	6 місяців
3	Відмова від складних технічних рішень на користь простоти впровадження	Мала ймовірність отримання ресурсів	3 роки

На основі аналізу можливих альтернатив було вирішено обрати підхід використання однієї екосистеми хмарних послуг для просто інтегрування між собою сервісів та покращення надійності їх взаємодії.

4.4 Розроблення ринкової стратегії проекту

Таблиця 4.14 – Вибір цільових груп потенційних споживачів

№ п / п	*Опис профілю цільової групи потенційних клієнтів*	*Готовність споживачів сприйняти продукт*	*Орієнтовний попит в межах цільової групи (сегменту)*	*Інтенсивність конкуренції в сегменті*	*Простота входу у сегмент*
1	Розробники застосунків	Висока готовність через високу потребу послугах, що пропонуються	Високий попит через необхідність	Внутрішня конкуренція через можливість розробників самотужки надавати собі ці послуги	Складність пов’язана з відсутністю зарекомендованого результату на ринку
2	Компанії-утримувачі магазинів застосунків	Готовність через потребу захищеності ринку застосунків	Високий попит через намагання забезпечити кращу безпеку користувачів маркету	Конкуренція незначна через можливість самостійного використання послуг	Складність полягає в відсутності потужної маркетингової інтеграції проекту
Обрано запропоновані цільові групи, поодинокі випадки домашніх розробок до уваги не бралися.					

Керуючись ринковою стратегією оптимальним варіантом було обрано стратегію диференційованого маркетингу.

Таблиця 4.15 – Визначення базової стратегії розвитку

№ п/п	*Обрана альтернатива розвитку проекту*	*Стратегія охоплення ринку*	*Ключові конкурентоспроможні позиції відповідно до обраної альтернативи*	*Базова стратегія розвитку*
1	Флангова атака	Стратегія диференційованого маркетингу	Сильні перспективи та якості запропонованого рішення	Стратегія диференціації

Таблиця 4.16 – Визначення базової стратегії конкурентної поведінки

№ п/п	*Чи є проект «першопрохідцем» на ринку?*	*Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?*	*Чи буде компанія копіювати основні характеристики товару конкурента, і які?*	*Стратегія конкурентної поведінки*
1	Частково рішення є унікальним	Пропонується поєднання перерахованих варіантів пошуку клієнтів	Копіювання полягатиме у підбиранні оптимальної стратегії послуг	Стратегія флангової війни

Таблиця 4.17 – Визначення стратегії позиціонування

№ п/п	*Вимоги до товару цільової аудиторії*	*Базова стратегія розвитку*	*Ключові конкурентоспроможні позиції власного стартап-проекту*	*Вибір асоціацій, які мають сформувати комплексну позицію власного проекту*
1	Забезпечення максимальної захищеності розроблюваного застосунку з метою залучення більшої аудиторії користувачів, швидке реагування на нові загрози мобільній безпеці, гнучке налаштування хмарних систем під власні потреби, можливість аудиту власних систем безпеки мобільних застосунків та їх коригування	Стратегія диференціації	Новизна підходу та універсальність методів надання послуг	Безпека мобільних застосунків, новітні пропозиції щодо інструментів захисту, гнучкий підхід до клієнтів послуг

4.5 Розроблення маркетингової програми стартап проекту

Таблиця 4.18 – Визначення ключових переваг концепції потенційного товару

№ п/п	*Потреба*	*Вигода, яку пропонує товар*	*Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити*
1	Захищеність мобільних застосунків	Популярність товару через захищеність користувацьких даних застосунку	Багаторівневий захист
2	Налагоджена підтримка захисту застосунку	Використання хмарних технологій	Висока надійність запропонованих сервісів, легка керованість та конфігурація
3	Швидке реагування на нові загрози	Максимальна захищеність застосунків від загроз нульового дня	Дистанційне впровадження хмарного захисту від нових загроз

Таблиця 4.19 – Визначення ключових переваг концепції потенційного товару

Рівні товару	*Сутність та складові*
1	2
I. Товар за задумом	Розробка системи безпеки для мобільного застосунку з використання запропонованої архітектури та хмарних сервісів

Продовження Таблиці 4.19

1	2		
II. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх /Тл/Е/Ор
	Захищеність мобільних застосунків	М	Вр/Тх/Тл
	Налагоджена підтримка захисту застосунку	М	Тх/Тл/Е/Ор
	Швидке реагування на нові загрози	М	Вр/ Тх /Тл/Е/Ор
	Якість: Забезпечення захищеності мобільного застосунку, універсальний захист від більшості відомих загроз		
	Пакет розробок, що інтегруються в застосунок		
	Марка: *SMSec*		
III. Товар із підкріпленням	До продажу: для покращення продажів існує варіант виїзних демонстрацій макетів роботи запропонованих сервісів		
	Після продажу: Проведення потужної рекламної кампанії		
За рахунок чого потенційний товар буде захищено від копіювання: право на інтелектуальну власність			

Таблиця 4.20 – Визначення меж встановлення ціни

№ п/п	*Рівень цін на товари-замінники*	*Рівень цін на товари-аналоги*	*Рівень доходів цільової групи споживачів*	*Верхня та нижня межі встановлення ціни на товар/послугу*
1	20000 ум.од	15000 ум.од	Високий	12000-15000 ум.од в залежності від ступеню інтеграції системи захищеності

Таблиця 4.21 – Формування системи збуту

№ п/п	*Специфіка закупівельної поведінки цільових клієнтів*	*Функції збуту, які має виконувати постачальник товару*	*Глибина каналу збуту*	*Оптимальна система збуту*
1	Необхідність забезпечення захищеності розроблюваних застосунків	Постійне покращення маркетингової політики, аналіз ринкових тенденцій та оновлення послуг відповідно до новітніх розробок	Дворівневий канал збуту з використанням дистриб'юторів та використанням сторонніх хмарних сервісів	Об'єднання з компаніями посередниками задля формування потужної системи збуту

Таблиця 4.22 – Концепція маркетингових комунікацій

№ п/п	*Специфіка поведінки цільових клієнтів*	*Канали комунікацій, якими користуються цільові клієнти*	*Ключові позиції, обрані для позиціонування*	*Завдання рекламного повідомлення*	*Концепція рекламного звернення*
1	2	3	4	5	6

Продовження Таблиці 4.22

1	2	3	4	5	6
1	Можливість аналізу та тестування пропонованих послуг з детальними подробицями функціонування різних рівнів захисту	Інтегровані маркетингові комунікації	Застосування хмарних технологій, гнучкість налаштувань послуг, постійна підтримка продукту, відсутність аналогів за спектром надання послуг	Популяція захищеності застосунків перед цільовою аудиторією	Реклама, яка презентує новітній підхід до якісного захисту приватних даних мобільних застосунків

Висновки до розділу 4

В даному розділі було запропоновано стартап проект, стратегія розгортання якого базується на детальному аналізі ринку споживачів – розробників застосунків, та високому попиті на безпеку приватної інформації користувачів, що популяризує застосунок як гарант безпеки. Враховуючи дослідження ринку даний проект має високі шанси виходу на ринок та подальше закріплення позицій, як потужного рішення захисту, що слідує тенденціям систем безпеки.

ВИСНОВКИ

У роботі було розглянуто проблему безпеки мобільних застосунків для різних мобільних платформ та рішень. У ході детального аналізу ринку мобільних застосунків було доведено привабливість використання мобільних продуктів, їх різноманітність монетизації та фактичну цінність для користувачів.

На основі уподобань та тенденцій розвитку мобільних платформ для детального вивчення було взято дві найбільші мобільні платформи iOS та Android та проаналізовано вже реалізовані методи та системи безпеки від розробників цих платформ.

Базуючись на проведеному дослідженні було запропоновано варіант механізму безпеки мобільного застосунку базуючись на проаналізованій архітектурі мобільних застосунків, їх призначенні та доцільності безпеки в цілому. Також було запропоновано систему захисту запропонованого мобільного застосунку, що складається з різноманітних технік та інструментів забезпечення захищеності даних, які залежать від платформи, а також є загальними для розробки мобільних систем в цілому. Також було перелічено необхідний список організаційних підготувань задля продумування захищеності системи.

Такий підхід повинен забезпечити необхідний рівень захищеності в мобільних застосунках. Належний захист необхідно організувати використовуючи комплексний підхід до проблеми, враховуючи всі відомі напрямки загроз.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1 «Mobile App Usage - Statistics & Facts» [Електронний ресурс]. — Statista – The portal for statistics. — Access mode: https://www.statista.com/topics/1002/mobile-app-usage/ .

2 «Топ-5 мессенджеров в Украине» [Електронний ресурс]. — AIN.UA. — Режим доступу: https://ain.ua/2018/04/10/top-5-messendzherov-v-ukraine/. — 10.04.2018р.

3 Указ Президента України Про рішення Ради національної безпеки і оборони України від 28 квітня 2017 року "Про застосування персональних спеціальних економічних та інших обмежувальних заходів (санкцій)". — Газета «Урядовий кур'єр» [Електронний ресурс]. — 28.04.2017. — Режим доступу: https://ukurier.gov.ua/media/documents/2017/05/16/2017_05_17_133upu.pdf.

4 Anna Washenko. «App store revenue to exceed $101B by 2020; music apps are key» [Електронний ресурс]. — RAIN News. — 11.02.2016. — Access mode: https://rainnews.com/app-store-revenue-to-exceed-101b-by-2020-music-apps-are-key/.

5 Gal Beniamini. BitUnmap: Attacking Android Ashmem [Електронний ресурс]. — December 1, 2016. — Access mode: https://googleprojectzero.blogspot.com/2016/12/bitunmap-attacking-android-ashmem.html.

6 iOS Security. iOS 12.1 [Електронний ресурс]. — November 2018. — Access mode: https://www.apple.com/business/site/docs/iOS_Security_Guide.pdf.

7 Heinrich Kersten, Gerhard Klett. Mobile Device Management [Електронний ресурс]. — Hüthig Jehle Rehm. — 2012. — 233 p.